Lecture Notes in Artificial Intelligence 5406

Edited by R. Goebel, J. Siekmann, and W. Wahlster

Subseries of Lecture Notes in Computer Science

Amba Kulkarni Gérard Huet (Eds.)

Sanskrit Computational Linguistics

Third International Symposium
Hyderabad, India, January 15-17, 2009
Proceedings

Springer

Series Editors

Randy Goebel, University of Alberta, Edmonton, Canada
Jörg Siekmann, University of Saarland, Saarbrücken, Germany
Wolfgang Wahlster, DFKI and University of Saarland, Saarbrücken, Germany

Volume Editors

Amba Kulkarni
University of Hyderabad, Department of Sanskrit Studies
Hyderabad 500046, India
E-mail: apksh@uohyd.ernet.in

Gérard Huet
INRIA, Centre de Paris-Rocquencourt
78153 Le Chesnay Cedex, France
E-mail: gerard.huet@inria.fr

Library of Congress Control Number: 2008942497

CR Subject Classification (1998): J.5, H.3.1, I.2.7, F.4.2

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-93884-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-93884-2 Springer Berlin Heidelberg New York

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12600052 06/3180 5 4 3 2 1 0

Preface

This volume presents the proceedings of the Third International Sanskrit Computational Linguistics Symposium hosted by the University of Hyderabad, Hyderabad, India during January 15–17, 2009. The series of symposia on Sanskrit Computational Linguistics began in 2007. The first symposium was hosted by INRIA at Rocquencourt, France in October 2007 as a part of the joint collaboration between INRIA and the University of Hyderabad. This joint collaboration expanded both geographically as well as academically covering more facets of Sanskrit Computaional Linguistics, when the second symposium was hosted by Brown University, USA in May 2008.

We received 16 submissions, which were reviewed by the members of the Program Committee. After discussion, nine of them were selected for presentation. These nine papers fall under four broad categories: four papers deal with the structure of Pāṇini's Aṣṭādhyāyī. Two of them deal with parsing issues, two with various aspects of machine translation, and the last one with the Web concordance of an important Sanskrit text.

If we look retrospectively over the last two years, the three symposia in succession have seen not only continuity of some of the themes, but also steady growth of the community. As is evident, researchers from diverse disciplines such as linguistics, computer science, philology, and vyākaraṇa are collaborating with the scholars from other disciplines, witnessing the growth of Sanskrit computational linguistics as an emergent discipline.

We are grateful to S.D. Joshi, Jan Houben, and K.V.R. Krishnamacharyulu for accepting our invitation to deliver the invited speeches.

The symposium hosted a traditional debate, called Vidvat Sabha, where participants presented views of different schools of language analysis – vyākaraṇa, nyāya, and mīmānsā – on the meaning of a sentence and debated on it. We thank all the participants of the Vidvat Sabha, in particular Prahlad Char, who accepted our invitation to chair the session. Since the Vidvat Sabha is a 'live program' to be watched and listened to, we do not have its report in these proceedings, however, the video recordings of this program will be available on the symposium website shortly.

We thank the University of Hyderabad, INRIA, Ministry of Information Technology, Government of India, Rashtriya Sanskrit Sansthan, and Sanskrit Academy for their valuable support.

This symposium would not have been possible without the involvement of several persons. In addition to the Program Committee members, we would like to thank J.S.R.A. Prasad and K. Subramaniam for their help in the local organization.

We thank all the researchers who responded to our call for papers and participants in this event without whose response the symposium and workshop would not have been a success.

Finally we thank the editorial team of Springer, who have provided us with a platform to record the proceedings of the symposia so as to make them available to other researchers and help in building the community right from its birth.

January 2009

Amba Kulkarni
Gérard Huet

Organization

SCLS 2009 was organized by the Department of Sanskrit Studies, University of Hyderabad, in joint collaboration with INRIA, France and the Sanskrit Academy, Hyderabad

Program Chair

Amba Kulkarni	University of Hyderabad, Hyderabad, India
Gérard Huet	INRIA, Rocquencourt, France

Steering Committee

Brendan Gillon	McGill University, Montreal, Canada
Gérard Huet	INRIA, Rocquencourt, France
Amba Kulkarni	University of Hyderabad, Hyderabad, India
Malhar Kulkarni	IIT Mumbai, India
Peter Scharf	Brown University, RI, USA

Program Committee

Stefan Baums	Department of Asian Languages and Literature, University of Washington, USA
Pushpak Bhattacharya	Department of Computer Science and Engineering, IIT, Mumbai, India
Brendan S. Gillon	Department of Linguistics, McGill University, Montreal, Canada
François Grimal	École française dExtrême-Orient, Pondicherry, India
Jan Houben	Directeur d'Etudes, École Pratique des Hautes Études, Paris
Malcolm Hyman	Max-Planck-Institut für Wissenschaftsgeschichte, Berlin, Germany
Girish Nath Jha	Special Centre for Sanskrit Studies, J.N.U. New Delhi, India
K.V.R. Krishnamacharyulu	Department of Vyākaraṇa, Rashtriya Sanskrit Vidyapeetham, Tirupati, India
Malhar Kulkarni	Department of Humanities and Social Sciences, IIT, Mumbai, India
Lalit Kumar Tripathi	Rashtriya Sanskrit Sansthan, Allahabad, India

Peter M. Scharf	Department of Classics, Brown University, Providence, RI, USA
Srinivas Varakhedi	Director, Sanskrit Academy, Hyderabad, India

Sponsoring Institutions

University of Hyderabad, India
INRIA, France
Ministry of Information Technology, India
Rashtriya Sanskrit Sansthan, Delhi, India
Sanskrit Academy, Hyderabad, India

Table of Contents

Background of the Aṣṭādhyāyī

S.D. Joshi

Retired Prof. and Head, Department of Sanskrit and Director, CASS, Pune, India

1. I have hesitated in accepting the invitation extended to me by Amba Kulkarni on September 9. The main reason was that I am not acquainted with what is called Sanskrit Computational Linguistics, or with theories of Machine Translation, or with information theory. In fact, I know nothing about these subjects. So what can I tell you? In view of my deeply regretted lack of knowledge regarding the subjects mentioned, I have decided to deliver a talk on a subject of which I have some experience, namely, Pāṇini's linguistic analysis as shown in his method of analysis, in the development of theoretical concepts and in the composition of the Aṣṭādhyāyī. Clearly, Pāṇini, in applying his linguistic analysis of the spoken Sanskrit of his days, has developed a number of theoretical concepts which can be used for the analysis of other languages also. That is an elementary insight which proved to be fruitful already in the 19^{th} century when linguistics and especially comparative linguistics were developed as separate branches of science in Germany and France. Reading statements about information coding in which Pāṇini is hailed as an early language code information scientist, I am reminded of the situation in the early sixties, after Chomsky had published his book on Syntactic Structures in 1957. Here Chomsky introduced a type of grammar called transformational generative grammar. It earned him a great of applause, globally, I may say. Then it dawned on linguists that Pāṇini had also composed a generative grammar. So Pāṇini was hailed as the fore-runner of generative grammar. That earned him a lot of interest among linguists. Many linguists, foreign as well as Indian, joined the bandwagon, and posed as experts in Pāṇinian grammar on Chomskyan terms. Somewhat later, after Chomsky had drastically revised his ideas, and after the enthusiasm for Chomsky had subsided, it became clear that the idea of transformation is alien to Pāṇini, and that the Aṣṭādhyāyī is not a generative grammar in the Chomskyan sense. Now a new type of linguistics has come up, called Sanskrit Computational Linguistics with three capital letters. Although Chomsky is out, Pāṇini is still there, ready to be acclaimed as the fore-runner of Sanskrit Computational Linguistics. I am, of course, grateful for the interest shown in Pāṇini.
2. So what to talk about? I can, obviously, refer to the 25 volumes published by the University of Pune, and the Sahitya Akademi, one series on sections of Mahābhāṣya and another series on sections of the Aṣṭādhyāyī. From the first series I expressly mention the Samarthāhnika, the Kārakāhnika, the Anabhihitāhnika and the Paspaśāhnika. In all of these books fundamental questions about Pāṇini's method of linguistic analysis have been discussed extensively. But references cannot make up a key-note address. So what I

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 1–5, 2009.

plan to do is to mention a number of typical features of the Aṣṭādhyāyī and some basic grammatical concepts applied in Pāṇini's analysis of the spoken Sanskrit of his days, and in the composition of the Aṣṭādhyāyī.

3. Pāṇini is short on theory, great on grammatical detail. A coherent linguistic theory can only be inferred from his detailed observations of linguistic data put in the form of rules. Questions of linguistic development, of historic sound change, and of history in general lie outside Pāṇini's interest.
4. Contrary to some Western misconceptions the starting point of Pāṇini's analysis is not meaning or the intention of the speaker, but word form elements as shown in the initial stages of the prakriyā. Here morphemic elements obtained from analysis are put side by side in an order of pūrva and para from left to right. Then by applying operations to these elements a derivation process starts. The process results in a word fit for use in vyavahāra, the every day usage of an educated brahmin .Thus we may say that Pāṇini starts from morphology to arrive at a finished word where no further rules become applicable. We have to bear in mind that Sanskrit is an inflecting language.
5. Is the Aṣṭādhyāyī rightly called a grammar? It certainly deals with the traditional parts of grammar in the West, namely, morphology, word-formation and syntax. On that account the name "grammar" is applied. It is, in fact, part of the title given by Böhtlingk to his edition of the Aṣṭādhyāyī. But the Aṣṭādhyāyī is not a grammar in this general Western sense of the word. It is a device, a derivational word-generating device. It presupposes knowledge of phonetics and it is based on morphemic analysis. It derives an infinite number of correct Sanskrit words, even though we lack the means to check whether the words derived form part of actual usage. As later grammarians put it, we are *lakṣaṇaikacakṣuṣka*, solely guided by rules. Correctness is guaranteed by the correct application of rules. For purposes of derivation as seen by Pāṇini a list of verbal bases, dhātus, is essential. That list is provided in the dhātupāṭha. It must have formed part of the Pāṇinian tradition from the very beginning.
6. Every ṡāstra 'branch of science' has its technical vocabulary. Technical terms require a definition of their meaning, as opposed to words in everyday speech which are characterized by free symbolization, not bound by a previous convention regarding meaning. The Aṣṭādhyāyī, being a śāstra, has its own technical vocabulary, consisting of saṁjñās 'technical terms' and pratyāhāras 'abbreviative designations.' The saṁjñās are usually, but not always, defined. The non-defined saṁjñās are borrowed from various other branches of science supposed to be generally known. I mention mantra, yajus, napuṁsaka, liṅga, kriyā, vartamāna, vibhakti, prathamā, jāti, dravya, guṇavacana, visarga, vākya, vidhi, samartha and upamāna. Use of pratyāhāras is made when the question is of enumerations of speech sounds or of suffixes. Pratyāhāras are an enumeration saving device.
7. Is semantics part of the Aṣṭādhyāyī? Or, put slightly differently, does meaning (artha) form part of Pāṇini's linguistic analysis? We have to be very careful here in what is understood by the word "meaning." In the Indian tradition artha is the thing-meant, the thing referred to, that to which

we refer by means of words and sentences. Taking artha in this sense, the answer to my question is, no. That is clearly stated by P1.2.56, *arthasya anyapramāṇatvāt* 'because artha is decided by something else (than the Aṣṭādhyāyī).' The idea is that the Aṣṭādhyāyī is no authority to decide that word A refers to item A and that word B refers to item B. That is decided by usage in which metaphor plays a big role. Obviously, this should not be taken to mean that lexical meaning is of no interest to the Aṣṭādhyāyī. The whole of the taddhita-section testifies to the opposite.

To specify the meaning in which a nominal form is used, its lexical meaning, Pāṇini uses meaning-conditions. They are usually stated in a locative nominal form, sometimes also by means of a phrase. I quote two examples. The first is P. 3.2.134. It prescribes the following kṛt-suffixes up to P. 3.2.177 in three meanings stated as tacchīla '(an agent) having such and such a habit,' taddharma '(an agent) having such and such a duty' and tatsādhukārin '(an agent) who does something well.' The second is P. 3.3.116. It deals with the kṛt suffix LyuṬ. The rule says *yena saṁsparśāt kartuḥ śarīrasukham* 'one account of contact with which the agent experiences a feeling of physical pleasure.'

In the taddhita-section the meaning-condition is often phrased by means of a pronominal form like tasya, tena followed by a noun or participle in the nominative. The whole serves as an adhikāra. But here also phrases may be used for the same purpose. I mention P. 4.2.59, *tad adhīte tad veda.*

8. Pāṇini's operational rules are generally substitution rules. Here the distinction between the original (sthānin) and the substitute (ādeśa) is essential. As far as further rule application is concerned, the substitute is declared to be like the sthānin (P. 1.1.56). An exception is made for rules which deal with the substitution of phonemes. An ingenuous idea of Pāṇini was to extend the concept of substitution to zero-substitution (lopa) also. Lopa is defined as adarśanam "disappearance from sight" (P. 1.1.60).
9. What about rule-order application in the Aṣṭādhyāyī? As is well-known, the Aṣṭādhyāyī has been divided into two parts, the siddha-kāṇḍa and the asiddha-kāṇḍa, the latter part starting from the rule *pūrvatrāsiddham* (P. 8.2.1). The asiddha-kāṇḍa is also known as the tripādī. In the earlier part rules are applied independently of the numerical order. In the tripādī rules are applied strictly according to their numerical order. Also, with regard to the application of a rule in the siddha-kāṇḍa a rule in the tripādī-section is asiddha. A rule A can be siddha '(regarded as) effected' or asiddha '(regarded as) non-effected' with regard to rule B in the sense that rule A is regarded as having taken effect before the application of rule B or not. Accordingly, rule B may become operative or not. This is a very useful grammatical fiction in the Aṣṭādhyāyī. The tripādī-section has been established to overcome difficulties in the random application order, when this order would lead to undesired results. The majority of rules put in the asiddha-section are rules dealing with consonant-substitutions due to sandhi.
10. Another situation in which the order of application of rules becomes vital is that of conflict (vipratiṣedha). The term vipratiṣedha has not been

defined in the Aṣṭādhyāyī, but it was taken up by Kātyāyana for explanation (vārtika I on P. 1.4.2). In the prakriyā a conflict may arise in the sense that two rules become applicable at the same stage. Here the question is of determining the stronger rule which is to prevail. Tradition, as embodied in Nāgeśa's Paribhāṣenduṡekhara, has formulated a number of principles to solve a conflict. I may point out that recently a considerable amount of work has been done on conflict-procedure, leading to the formulation by myself and P. Kiparsky of the siddha-principle. I won't bother you with further details on this intricate subject, but refer you to Vol. IV in the Aṣṭādhyāyī of Pāṇini Series, 1995, Introduction, p. viii-xi, Here the new ideas on the subject have been explained.

11. Kātyāyana, in the opening vārtika of the Mahābhāṣya, says *atha śābdānuśāsanam* 'now starts the instruction in words.' But what are words? Patañjali explains in his bhāṣya that words may belong to ordinary speech or the Veda. They are laukika or vaidika. Examples for both categories are quoted. Then he asks the question, in gauḥ what is the word (śabda)? The answer is that from this word we understand an object with a dewlap, a hump, hoofs and horns. Apparently, a word is that from which we understand a meaning in the sense of a thing-meant.

    Pāṇini's answer to the question what is a word is rather different and rather more linguistically precise. First of all, for "word" he does not use the word śabda, but he uses the term pada. Then he defines that term as *suptiṅantam* 'ending in a suP-suffix or in a tiṄ-suffix' (P. 1.4.14). Thus pada does not just mean "word". It means a fully derived word according to Pāṇinian standards. Clearly here Pāṇini does not enter into questions of meaning, but talks in terms of word form categories. The suffixes mentioned are listed by P 3.4.78 and P. 4.1.2. We further note that the endings called tiṄ are excluded from the designation kṛt (P. 3.1.93).

12. The derivational process, prakriyā, starts from a dhātu, a verbal base, a list of which is provided in the dhātupāṭha. What comes next in the derivation are suffixes (pratyayas), divided into kṛt and taddhita. The section dealing with the addition of suffixes starts from P 3.1.92, dhātoḥ. This is the central rule in the Aṣṭādhyāyī for purposes of derivation. The order of dhātu and pratyaya is fixed by P. 3.1.2, which says that a suffix is a following element. The derivational base of a subanta pada is either a dhātu + a kṛt suffix, which forms a nominal base, or a nominal base + a taddhita suffix, or a combination of nominal bases called samāsa. All of these derivational nominal bases are called prātipadika (P. 1.2.46). Thereafter a feminine suffix may be added to indicate feminine gender, and the suP-suffix comes to take care of gender other than the feminine and of number, and of case. The last two general stages of the derivation are reserved for the application of sandhi-rules and of accent-rules. We have to bear in mind that Sanskrit is a pitch-accented language, although, unlike in the Vedas, accent in Sanskrit is not indicated. Accent is treated by Pāṇini in great detail; from P. 6.1.158 to 6.2.199, in all 263 sūtras, with two isolated rules at the end of pāda 8.4. That is in short how the derivation of a nominal form goes, the whole process being

regulated by rules. As everybody knows, for some Sanskrit subanta words a derivational base is not reasonably available. They are declared to be avyutpanna 'underivable,' or they may be still be derived with the help of an ad hoc invented suffix.

One more point about prakriyā which may be of interest to you being computer-linguists. The Aṣṭādhyāyī is not just an analysis of what he calls bhāṡā, and what was called Sanskrit later on. It is also a generative calculus, which is actually the main thrust of the Aṣṭādhyāyī. Whereas the type of grammar developed in Greece and Rome is paradigmatic, the Aṣṭādhyāyī is a generative calculus known as prakriyā for which Bhaṭṭojī Dīkṣita composed the authoritative handbook known as the Siddhāntakaumudī. Mastery of Pāṇini is shown in mastery of prakriyā, and the rest is silence. The prakriyā evolves by means of rule operations in successive stages. This is strongly reminiscent of a mathematical procedure known as algorithm. Here the answer to a problem belonging to a class which has an infinite number of members is produced in a finite number of steps. As you undoubtedly know, in principle the calculus can be produced by a machine provided with a tape. That was shown already in 1937 by Turing. Thus, I think, we may say that Pāṇini whom I date around 350 B.C. has intuitively used this idea of calculus.

13. What about case, one may ask. The technical term in the Aṣṭādhyāyī is kāraka, literally "one who or that which brings about", introduced by P. 1.4.23. A satisfactory English translation is not found. Kāraka is a syntactic category, since it deals with the formal characteristics of word meaning combination according to the speaker's intention, whether in a word group or in a sentence. Kāraka is not a semantic category, nor a semantic-syntactic category which merely confuses the issue. For an exhaustive discussion of the grammatical points involved I may refer to the Kārakāhnika, published by the University of Poona in 1975.
14. Finally, I want to say something very briefly about Pāṇini's idea of vākya. The term is not defined in the Aṣṭādhyāyī. Literally the term means "what can be spoken", in distinction from vācya. The term is used in the sense of "utterance" whose end is marked by a pause (avasāna, P 1.4.110), but also in the sense of what we call a word group or sentence. Since Pāṇini uses the term *vākyādeḥ* 'at the beginning of a vākya' in P. 8.1.8, he must have had an idea where the vākya starts. In fact, it starts after a pause in speech. That is why Pāṇini need not define vākya and that has saved him a lot of trouble. The first attempts to formally define vākya stem from Kātyāyana. He has provided two definitions in the vārtikas. IX and X on P. 2.1.1.

September 24, 2008

Pāṇini's Grammar and Its Computerization: A Construction Grammar Approach*

Jan E.M. Houben

École Pratique des Hautes Études (SHP), Paris

Abstract. This article reviews the impact of modern theoretical views on our understanding of the nature and purpose of the grammar of Pāṇini (ca. 350 BCE), and argues that new possibilities for progress open up for our understanding of this ancient grammar by confronting it not with the presuppositions of generative grammar as has been done – with undeniable but limited theoretical profit -- in the last few decades, but with recently developed theories of construction grammar and cognitive linguistics. This, in turn, provides new perspectives on old problems in the study of Pāṇinian grammar, and especially on the challenge of its computerization. The present article focuses on general technical aspects of Pāṇini's grammar and is the counterpart of a recent study on the earliest available elaborate theory of Pāṇini's grammar, the one formulated by the grammarian-philosopher Bhartṛhari (5th cent. CE).

Keywords: Pāṇini's grammar, (transformative) generative grammar, construction grammar, cognitive linguistics, computerization of Pāṇini's grammar, levels of representation in Pāṇini's grammar, Dhātu-pāṭha, lists of roots.

1 Introduction

The history of grammatical thought in India can be estimated to be at least around 3000 years old, as we find hints to the analysis of verbal roots from various linguistic forms, finite verbs and nominal forms, in the Atharva-veda, and especially in the Brāhmaṇas (Liebich 1919, Palsule 1960). These three millennia of Indian grammatical thought have been dominated by Pāṇini's grammar for more than two thirds of the time, since the date of its composition, ca. 350 B.C.E.[1]

* Because of the limited time available for writing this article I have to refer to earlier publications (Houben 1999, 2003, 2006, 2008a, 2008b) for the substantiation of some of my points with detailed examples from the works of Pāṇini and Pāṇinīyas. A brief discussion of Pāṇini and his predecessors and successors, not only in their intellectual but also in their social and cultural contexts, is given in Houben 1997.

[1] Pāṇini's *rūpya* (A 5.2.120) refers to a type of coin which appeared in the Indian subcontinent only from the 4th century B.C.E. onwards: cf. von Hinüber 1989: 34 and Falk 1993: 304. The date of "ca. 350 B.C.E." for Pāṇini is thus based on concrete evidence which till now has not been refuted.

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 6–25, 2009.

As the earliest major grammatical description, Pāṇini's grammar is remarkably extensive in covering its object, surprisingly efficient and brief in formulation and presentation, and of impressive quality. Even then, it was marginally amended and improved upon in a long tradition, and on a large scale it was recast and abbreviated. While it never received a definitive replacement, numerous alternative grammars have been composed which adopted a great number of the techniques and materials of Pāṇini's grammar while modifying it – in the respective authors' view, improving on it – in other respects. The domination of Pāṇini's grammar over the practice of Indian grammar and Sanskrit literature can therefore be described as a kind of extended love-hate relationship.

The object of Pāṇini's grammar is (a) the language of the Vedic texts and (b) the current language of Pāṇini's time, which is very close to the "classical" Sanskrit that got established in subsequent centuries. The sophisticated and highly complex system of Pāṇini's grammar consists of the following components : (i) an inventory of phonemes in the form of fourteen formulas, the *pratyāhāra-sūtra*s; (ii) the grammatical rules or *sūtra*s, in eight books, collectively the Aṣṭādhyāyī (A); (iii) lists of roots or *dhātu*-s divided in ten major groups, collectively called the Dhātu-pāṭha (DhP); (iv) a number of lists of forms that are not derivable from roots, collectively the Gaṇapāṭha (GP); additional components that can be left out of consideration in a brief overview are (v) the *uṇādi-sūtra*s referring to suffixes that form nominal stems apart from the *kṛt*- and *taddhita*-formations that are extensively discussed in the Aṣṭādhyāyī; (vi) the *phiṭ-sūtra*s on the accents of derived forms; and (vii) the *liṅgānuśāsana* giving lists and rules to determine the gender of various words (according to A 1.2.53, knowledge of the gender of words can be presupposed and need not be taught in the grammar).

Grammars which present themselves as independent, even when they use many of the techniques and devices of Pāṇini, normally concern Sanskrit but also Pali or the closely related Prakrits.

A direct view on Pāṇini's grammar as composed and intended by the author and as accepted by first-generation users in the author's own time is impossible. The cultural and technical conditions of the transmission of knowledge in the Indian world – which, until several centuries after Pāṇini, was initially dominated by orality and later on by manuscript-literacy – allow us to achieve only a view that is to an important extent mediated. Three major steps in this mediation over many centuries can be distinguished, out of which the crucial importance of the last two, b and c, has been almost entirely neglected. The first step is (a) the interpretations and constructions of early grammarians whose work is sufficiently transmitted and whose thought concerns more or less the entire grammar of Pāṇini: Patañjali, 2nd cent. B.C.E., author of the Vyākaraṇa-Mahābhāṣya; Bhartṛhari, 5th cent. C.E., author of the Mahābhāṣya-dīpikā and of the Vākyapadīya, an investigation of theoretical and philosophical issues regarding basic concepts in Pāṇini's grammar; Vāmana and Jayāditya, 7th cent. C.E., (regarded as) joint authors of the Kāśikā; (b) the interpretations and constructions of "later" grammarians who perceive the nature and role of Pāṇini's grammar in specific ways in function of their study of the transmitted texts *and* in function of

the cultural and sociolinguistic conditions of their own time – partly similar, partly different from the conditions in Pāṇini's time; the view and constructions of later grammarians are all the more important because we know that the oral tradition knew important discontinuities at an early stage – referred to by the 5th century grammarian-philosopher Bhartṛhari – and that the written transmission depends on manuscripts whose physical lifespan is limited to around two to four hundred years, and hence on regular copying; (c) the interpretations and constructions of "western" scholars (and Indian scholars following the methods of modern linguistics) of Pāṇini's grammar who perceive the nature and role of Pāṇini's grammar in specific ways in function of their study of the available transmitted texts *and* in function of the nature and roles of grammars in "western" context, and of their own theoretical views on grammar and language – on the nature of words, nouns, verbs and the sentence – whether implicitly accepted or explicitly formulated. With regard to the highly sophisticated Indian sciences and disciplines pertaining to language, it has been rightly pointed out that it is difficult for modern scholars to detect and appreciate something in these linguistic works if they do not have already discovered it by themselves (Staal 1988: 47).

It is with regard to step (c) in our mediated view on Pāṇini's grammar that the presuppositions of construction grammar are of direct relevance. Construction grammar, sometimes abbreviated as CxG, refers to a "family" of theories or models of grammar that have started to attract wider attention especially since around 2000, when theories of the "family" of Chomskian transformational generative grammars were losing their attraction. Perhaps unexpectedly, the presuppositions of construction grammar also have implications for steps (a) and (b). Presuppositions of construction grammar overlap to a great extent with those of cognitive linguistics. Cognitive linguists investigate basic psychological mechanisms underlying all cognitive domains including the learning and use of language, normally without postulating an identifiable structure given before hand in language or in the language user as in Chomskian theory.

In several significant respects, a mediated view on Pāṇini's grammar in the light of construction grammar turns out to be different from a view on Pāṇini's grammar in the light of transformative generative grammar or of generative grammar. Moreover, it opens new perspectives on the computerization of this grammar. Because of the importance and authoritative status of Pāṇini's grammar in Indian cultural and literary history the great challenge of computerizing this grammar has attracted several scholars but till now no comprehensive and convincing results can be cited.

One of the problems in our understanding of and dealing with Pāṇinian grammar is that it has come to us without a statement of underlying theoretical views by the author himself. In the tradition of Pāṇinian grammar we do have quite elaborate theoretic and philosophical discussions of basic grammatical concepts in the work of Bhartṛhari, especially in his Vākyapadīya. Major presuppositions of Bhartṛhari, fortunately or unfortunately, do not match major presuppositions of Chomskian transformative generative grammar or those of generative grammar (Houben 2008b). Since our views on Pāṇini's grammar have

been very much informed, explicitly or implicitly, by theories of generative grammar it was till now not possible to see Bhartṛhari as a thinker developing a valid view on Pāṇini's grammar. Instead he has been regarded as someone carrying his readers away from grammar to a peculiar, idiosyncratic philosophy which does not fit very well in any of the traditional philosophical schools of Bhartṛhari's time, and which is hardly relevant to modern linguistic concerns.

In a recent study (Houben 2008b), I confronted foundational assumptions of cognitive linguistics with features of Bhartṛhari's theory of grammar and found, surprisingly, that in this light Bhartṛhari's theoretical investigations are of direct relevance to current linguistic concerns, and, moreover, that he develops a valid and directly relevant theoretical perspective on Pāṇini's grammar, in spite of the eight to nine centuries that intervene between him and Pāṇini (which is at least 15 centuries less than those intervening between us and Pāṇini). The present article is a counter-part to this article on "Bhartṛhari as a cognitive linguist" as it explores the relevance of three foundational assumptions of construction grammar (which, as said, partly overlap with presuppositions of cognitive linguistics) for Pāṇini's grammar as known to us. To make this article comparable and compatible with the Bhartṛhari article I will refer to the same lists of foundational assumptions, the list for construction grammar and the list for cognitive linguistics, that were used in that article.

One major difference between the two articles is that in the case of Bhartṛhari and cognitive linguistics we can directly match (or contrast) theory and theory, whereas in the case of Pāṇini and construction grammar we have on the one hand Pāṇini's full-fledged grammar and on the other hand the theories of construction grammar which have been used with regard to problems of language learning and language use but, to my knowledge, it has not yet led to the formulation of a comprehensive grammar entirely on the basis of these theories. In our present study we will therefore explore to what extent principles of construction grammar *can* be assumed to be underlying the grammar of Pāṇini as we have it. It will force us to rethink certain views on Pāṇini's grammar that have till now seemed entirely natural and indisputable. It will force us also to rethink some of the currently indisputed choices to emphasize some of the relevant ancient and pre-modern texts and to neglect others. It may provide new perspectives on how Pāṇini's grammar originated and how it was used, which also implies a new perspective on what deserves to be central and what secondary in its computerization.

In recent years William Croft has argued in favour of what he calls Radical Construction Grammar (e.g., Croft 2001, 2003, in prep.), in contradistinction to conventional construction grammar, which he labels "vanilla construction grammar". The aims of Croft include the comparison of constructions in different languages, which is not relevant in the case of Pāṇini's sanskrit grammar. According to Croft, three (Croft 2003) or four theses (Croft in prep.) are accepted by conventional construction grammarians, whereas his own Radical Construction Grammar accepts a few more theses which emphasize that what the first theses describe as conventional construction grammar is "all that is

universal in formal syntactic representation” (2003: 4). For the present purpose, we can limit ourselves to Croft’s first four theses, supposed to be valid for most theories of construction grammar:

(1) The basic unit of grammatical representation is a pairing of form and meaning, where the form may range from the complex and schematic to the atomic and substantive.
(2) The basic units of grammatical representation are symbolic, that is, for a grammatical unit there is no separation of (a) the form and (b) the meaning or function of that form.
(3) According to Croft's third thesis, the constructions of a language form a structured inventory.
(4) According to the fourth thesis which we find in Croft (in prep.), usage is the basis of constructions.

For the sake of reference I will give here also the list of foundational assumptions formulated by Adele E. Goldberg in 1996 which I used in the Bhartṛhari article. Although in the title of her article she speaks of “construction-based grammar” the list is said to represent “widely shared foundational assumptions of cognitive linguists.”

1. Semantics is based on the speaker's *construals* of situations, not on objective truth conditions (Langacker 1985, 1987, 1988; Fauconnier 1985; Lakoff 1987; Talmy 1985).
2. Semantics and pragmatics form a continuum, and both play a role in linguistic meaning. Linguistic meaning is part of our overall conceptual system and not a separate modular component (Talmy 1978, 1985; Haiman 1980; Lakoff 1987; Langacker 1987)
3. Categorization does not typically involve necessary and sufficient conditions, but rather central and extended senses (Rosch 1973; Rosch et al. 1976; Lakoff 1977, 1987; Haiman 1978; Fillmore 1982; Hopper and Thompson 1984; Givón 1986; Brugman 1988; Taylor 1989; Corrigan et al. 1989)
4. The primary function of language is to convey meaning. Thus formal distinctions are useful to the extent that they convey semantic or pragmatic (including discourse) distinctions (Wierzbicka 1986, 1988; Lakoff 1987; Langacker 1987; Haiman 1985; Croft 1991; Deane 1991)
5. Grammar does not involve any transformational component. Semantics is associated directly with surface form.
6. Grammatical constructions, like traditional lexical items, are pairings of form and meaning. They are taken to have a real cognitive status, and are not epiphenomena based on the operation of generative rules or universal principles (Fillmore et al. 1987; Lakoff 1987; Wierzbicka 1988; Goldberg 1995)
7. Grammar consists of a structured inventory of form-meaning pairings: phrasal grammatical constructions and lexical items (Fillmore and Kay 1993; Lakoff 1987; Langacker 1987; Wierzbicka 1988; Goldberg 1995).

Foundational assumptions of construction grammar and of cognitive linguistics are usually formulated in contradistinction to those of generative grammar or transformative generative grammar. Pāṇini's grammar, however, does have parts and aspects that are very well addressed in the light of generative grammar or transformative generative grammar. Other basic and crucial aspects, however, are destined to remain un-recognized and unexplored if the family of generative grammars form our only theoretical frame of reference.

2 Construction Grammar and Pāṇinian Grammar

According to Croft's first thesis, "the basic unit of grammatical representation is a pairing of form and meaning, where the form may range from the complex and schematic to the atomic and substantive," in other words, from phrase structures and idioms to words and morphemes. This refers to the syntax – lexicon continuum, which Goldberg addressed in foundational assumption no. 6 "Grammatical constructions, like traditional lexical terms, are pairings of word and meaning." The result is that for construction grammarians the lexicon becomes an inventory of lexical items in the classical sense as well as constructions and even lexically unfilled constructional idioms. The counterpart to this thesis is found in the family of generative grammars which typically distinguish and separate different components in the grammar, mainly a lexicon, or lists of lexical items, and an (autonomous) syntax, or a body of general rules.

Langacker (2000 : 2) refers to this as the Rule / List fallacy, which implies "the spurious assumption that rules and lists are mutually exclusive." According to Langacker, this fallacy should be avoided by including in the grammar

> both rules and instantiating expressions. This option allows any valid generalizations to be captured (by means of rules), and while the descriptions it affords may not be maximally economical, they have to be preferred on grounds of psychological accuracy to the extent that specific expressions do in fact become established as well-rehearsed units. Such units are cognitive entities in their own right whose existence is not reducible to that of the general patterns they instantiate. (Langacker 2000 : 2)

In the practice of grammar, the separation of syntax and lexicon can therefore be overcome either by setting up a lexicon that includes idioms, phrase structures, etc., or by including lists of lexical items in the syntax. This is precisely the situation we find in Pāṇini's grammar : the grammar contains numerous lists integrated into the rules, and moreover a number of major lists in the form of roots and nouns assorted in sophisticated ways.

To Langacker's remarks we should add that the aim to have an accurate description of psychological processes underlying the use of language is shared with the generative grammars which claim that the division of grammar into components reflects the human capacity for learning and using language. In

classical transformational generative linguistics it is the syntax which forms the core of a postulated universal Language Acquiring Device (LAD). This explains the fascination of generative linguists with the syntactic rules as the central component of grammar. As Kiparsky (2002: 1) observed :

> Generative linguists for their part have marveled especially at its ingenious technical devices [in use in the body of rules (JH)], and at [the] intricate system of conventions governing rule application and rule interaction that it presupposes, which seem to uncannily anticipate ideas of modern linguistic theory (if only because many of them were originally borrowed from Pāṇini in the first place).

Of the theoretical aim of somehow capturing universal psycho-linguistic patterns in the grammar there is no trace either in Pāṇini's grammar or in the theoretical discussion of Bhartr̥hari. On the contrary, Bhartr̥hari argues that the divisions accepted in grammar are for the sake of analysis and description only and have no absolute status, and that, for instance, in the understanding of a sentence by a language user there is no definitive status of the parts of a sentence, which each individual may provisorily isolate in his own way:

> *arthaṁ kathaṁ cit puruṣaḥ   kaś cit saṁpratipadyate /*
> *saṁsr̥ṣṭā vā vibhaktā vā   bhedā vākyanibandhanāḥ //*
> A person understands a meaning in one way or the other.
> Whether combined or separated, parts are based on the sentence.
> (Vākyapadīya 2.39)

With regard to the study of Pāṇini's grammar, however, there is a risk that the perspective of generative linguistics leads not only to a fascination with the body of rules but also to a neglect of other aspects of the grammar or to a tendency to see the other components as both separable from and secondary to the body of rules.

Conversely, the perspective of construction grammar invites us to re-valuate the lists, especially the most sophisticated lists of assorted roots, the Dhātupāṭha, in which we find stored much grammatical information on each root. The postulation of a root as the element underlying numerous verbal and nominal forms actually occurring in the language is in each case a grammatical achievement. For the Dhātu-pāṭha presupposed in his grammar, Pāṇini was indebted to generations of previous grammatical thinkers, from the time of the Atharva-veda and Brāhmaṇas onwards. The fact that the current Dhātupāṭha contains *dhātusūtra*s, rules specifically applicable to a set of roots, and that through their categorization and through markers in the form of accents and labels in the form of phonemes a root evokes specific sets of rules in the body of rules or Aṣṭādhyāyī suggests the validity of a view on the grammar of Pāṇini that is an inversion of the common view on this grammar (and an inversion of the generative linguist's view): the rules appear as an appendix to the lists of roots, rather than the lists of roots being appendices to the body of rules. The Aṣṭādhyāyī and perhaps its predecessors thus appear as integrations of separate

sets of rules, some of which concern specific sets of assorted roots (others being concerned with sandhi-rules, etc.). Moreover, from the point of view of a grammar user of Pāṇini's own time, the analysis of whose conditions has remained surprisingly poor in the generative linguist's framework, the selection of a suitable root is normally the starting point of the synthetic part of his consultation cycle.

According to Croft's second thesis, "the basic units of grammatical representation are symbolic, that is, for a grammatical unit there is no separation of the form and the meaning or function of that form." This amounts to an entailment of Goldberg's foundational assumptions 4 and 6: grammatical constructions do not have an independent formal status, nor do meaning and function resort to a separate component of the grammar. In Croft's formulation the thesis includes the acceptance of a continuity of semantics and, what Croft calls, "conventional discourse or information structural properties" (2003: 3). This is Goldberg's foundational assumption 2, the continuity of semantics and pragmatics.

In this perspective it is "wrong" – or: it is a theoretical exercise more inspired by modern theoretical concerns than by ancient practice or theory of grammar – to postulate a level of "pure" semantics, and even more "wrong" to suggest that this level of "pure" semantics is the starting point for uni-directional derivations in Pāṇini's grammar. In an earlier article on "meaning statements in Pāṇini's grammar" (Houben 1999) I discussed the views formulated in Kiparsky and Staal (1969), Bronkhorst (1979), Joshi and Roodbergen (1975) and Kiparsky (1982) according to which "semantics" or "meanings" form the starting point of the derivation of words in Pāṇini's grammar. Also in his lectures on the architecture of Pāṇini's grammar (Kiparsky 2002: 2-6), Kiparsky sticks to the postulation of a first level of "semantic information" in Pāṇini's grammar. This is all the more problematic as Kiparsky also postulates that "The grammar is a device that starts from meaning information such as [5] and incrementally builds up a complete interpreted sentence," where [5] refers to a case where, basically, kārakas are assigned on the basis of "semantic information."

This is not that much different from Kiparsky and Staal (1969), except that in this earlier article the formulation leans more to Chomskian generative grammar. As I argued extensively in 1999, the view that Pāṇini's grammar is a device "to encode a given meaning and to produce an expression" is untenable: "how the semantic level can be placed at the basis and, as far as derivations are concerned, at the beginning of the sophisticated grammar of Pāṇini, while it is admitted at the same time that this semantic level is very sketchy" (Houben 1999: 26-27). Criticizing the partly parallel view of Bronkhorst according to which "meaning elements" are the input of Pāṇini's grammar I observed similarly: "Just as a semantic level with sketchy representations of semantic relations can hardly be accepted as forming the basis and starting point of Pāṇini's grammar, in the same way the terms which Bronkhorst considers to be Pāṇini's 'semantic elements' are too vague and insufficient to initiate the procedures of Pāṇini's grammar and to direct them with precision to the desired

utterances" (Houben 1999: 29). The appropriateness of my refusal to accept "pure" meanings or "pure semantics" as a significant level or stage in Pāṇini's grammar, for which no direct traditional support exists, finds support in this basic thesis of construction grammar: for a grammatical unit there is no separation of (a) the form and (b) the meaning or function of that form.

If "pure" meanings or "pure semantics" are not the starting point of the derivations in Pāṇini's grammar, then what is the starting point? As argued in 1999 and 2003, we have to understand the nature and purpose of Pāṇini's grammar in its specific context which is quite different from that of modern grammars. Strictly speaking it is not incorrect to say, with Kiparsky (2002) that "Pāṇini studied a single language"; however, this statement is incomplete on a vital point: Pāṇini was definitely aware of various "substandard" forms of the language, forms which from a modern perspective we would assign to an altogether different language such as Prakrit. The system of Pāṇini's grammar "clearly requires a user who wants to check and possibly improve a preliminary statement" (Houben 2003: 161). The system implies the presence of a knowledgeable user, a preliminary statement, and the application of first analytic and next synthetic procedures to the words in it, with the user keeping in mind the preliminary statement and its purport, and aiming at the best possible, *saṁ-skr̥ta* form of his preliminary statement.

The concrete starting point for a derivation in the synthetic phase of the consultation cycle of a user of grammar in Pāṇini's time will then never be "pure" meaning or an autonomous level of semantic representations but the selection of a root – for instance, *bhū* 'to be' – or a form from lists of underived stems, pronominal forms, etc., in which form and meaning are inseparably integrated. In the sociolinguistic context of Pāṇini's time we can suppose that the preliminary statement of the user of the grammar contained not necessarily only "perfectly formed" words but also substandard ones, for instance *honti* or *bhonti* instead of *bhavanti*. The knowledge of the user of grammar in Pāṇini's time concerns not only the basic outlines of the grammar and knowledge of the language aimed at, but also substandard forms current in his time and area.

What does this mean for the four "levels of representation" in the derivation of forms postulated by Kiparsky and Staal in 1969 and confirmed with minor modifications by Kiparsky in 2002? Against the background of then current generative grammar theories of "deep structure" in linguistic utterances, the scheme of four levels of representation seemed attractive in 1969. An opposite trend is visible in construction grammar, as testified for instance in Goldberg's fifth thesis: "Grammar does not involve any transformational component. Semantics is associated directly with surface form." With regard to the first part of this statement, formulated explicitly in opposition to transformational generative grammar: it became soon clear to scholars, how ever much they were inspired by transformational generative linguistics, that the presence of syntactic transformations (for instance, from passive to active constructions, etc.) cannot be accepted for Pāṇini's system. The second part of Goldberg's statement is what also appears to be the desired outcome of the present thesis of Croft: "no

separation of the form and the meaning or function of that form." Even a little familiarity with Pāṇini's system, however, will make it clear that, how ever much one may be inspired by construction grammar or cogntive linguistics, at least two distinct levels of derivation are to be accepted: a level of morphological representations (where we find roots, stems, suffixes) and a level of phonological representations (with words in their final form after the application of all substitution rules including those of sandhi).

Is any other level to be accepted? It turns out to be the case that no additional level of representation is needed to account for Pāṇini's system. Above we have already dispensed with a level of "pure" semantic representations, as its postulation is untenable. In an earlier article (Houben 1999), when the potential usefulness of construction grammar had not yet attracted my attention, I proposed to replace Kiparsky's (and Kiparsky's and Staal's) level of semantics with a level of "semantics, pragmatics and intentionality," and I emphasized its unformalizable nature, which seems quite disastrous from the perspective of generative linguistics, but which at the end only means that we need a knowledgable user of the grammar, familiar with the language and basic outlines of the grammar, and also a preliminary statement that is the starting point of the consultation cycle. One more level remains in Kiparsky's scheme, that of "morphosyntactic representation," earlier referred to as "abstract syntax (e.g., kārakas)". Even from Kiparsky's own account, e.g. his recent one of 2002, it is clear that this is in fact not an autonomous level of representation. I would now like to propose that both this and the "level" of "semantics, pragmatics and intentionality" are better regarded as domains of consultation, which allow the user of the grammar to label the linguistic forms of his preliminary sentence according to syntactically relevant categories of meaning or according to semantically relevant generalizations of form (suffixes). The proof of the validity of this scheme of the architecture of Pāṇini's grammar is provided by Kiparsky's own account of his four levels of representation (2002). Although, as we have seen, according to his explicit statement, "The grammar is a device that starts from meaning information ... and incrementally builds up a complete interpreted sentence" (Kiparsky 2002: 4), Kiparsky defeats his own account by placing the "output" of the correct sentence at the beginning. After giving his scheme of four levels of representation under [1], his immediate next step is:

"Consider the sentence whose output (phonological) form is shown in [2]:

[2] *vánād grā́mam adyópétyaudaná āśvapaténāpāci*[2]

'When Āśvapata came from the forest to the village today, he cooked some rice.' "

It is difficult to find a better confirmation of my thesis (Houben 1999, 2003) that not a semantic level but a preliminary utterance forms the starting point of a derivation according to Pāṇini. That the two "broad classes of rules" which should "effect a mapping" between the first and second and the second and third

---

[2] Kiparsky (2002 : 3) gives the last part of the sentence as : *āśvapaténā́pāci*, omitting the application of A. 8.1.28 *tiṅ atiṅaḥ*.

level do not concern an autonomous first and second level of representation is moreover clear from the way these classes of rules are referred to in Kiparsky's scheme. The first class of rules would effect the "assignment of kārakas and of abstract tense": but to what are these kārakas and abstract tenses (laṭ, etc.) assigned? Not to the semantic representations of level one, but to the words of the preliminary utterance, in accordance with my thesis and as *de facto* demonstrated by Kiparsky. Similarly, the "morphological spellout rules" which would effect a mapping between the level of morphosyntactic representation and that of abstract morphological representation is not sufficiently steered by the information available on the first two level, without taking into account a preliminary sentence, which is what Kiparsky actually does.

According to Croft's third thesis, "the constructions of a language form a structured inventory." This corresponds to Goldberg's foundational assumption 7: "Grammar consists of a structured inventory of form-meaning pairings: phrasal grammatical constructions and lexical items."

The negative implication of this thesis is that it takes away the theoretical basis for a grammar consisting in a pure and autonomous syntax to which lists of lexical items are appended. It also takes away the theoretical basis for a structure that is given before hand, whether in the Saussurean sense or in a more dynamic Chomskian sense (cf. Kaldewaij 1986). Since in Pāṇini's grammar we have only the grammar without direct statement of the underlying linguistic view, it is difficult to confirm directly whether this thesis is congenial to Pāṇini's approach or not. There is in any case no trace that a structure given before hand in language was accepted by Pāṇini or his predecessors. In the case of Bhartr̥hari's linguistic views, however, it is clear that they leave no room for the presence of a "structure given before hand" in Sanskrit, inspite of what one might expect on the basis of the oft-cited words of Sir William Jones (1786): "The Sanskrit language, whatever may be its antiquity, is of a wonderful structure."

The positive side of this thesis, as discussed by Croft, is that the inventory is widely characterized as a network. But he adds that the nature and structure of this network is a matter of debate, with as one of the parameters the extent to which inheritance and usage play a role in the formation of this network. The topic of "usage" appears again in the next thesis.

According to a further implication of this thesis, as it is the constructions that are the primitive elements of syntactic representation, grammatical categories such as "noun," "verb," etc., are derived from these. Bhartr̥hari must definitely be counted among those who would agree to this. In book 2 of his Vākyapadīya, verses 344-345, for instance, Bhartr̥hari refers positively to the view of another authority, Audumbarāyaṇa, according to whom the division into four categories of words disappears both in front of the mental nature of the sentence (the fact that it is based in the mind) and in front of the purposeful employment of language in daily life; both in the discipline of grammar and in daily life, however, we speak about language in terms of divided words and categories of words as this is convenient and widely applicable. This would further imply that, "the only internal syntactic structure of constructions is their meronomic structure

(i.e. the part-whole relation defined by an element's role in a construction), and symbolic links to components of semantic structure" (Croft, in prep.). This is again entirely congenial to Bhartṛhari's approach to language and grammar.

Would Pāṇini accept this too? We do not have direct access to the way grammatical concepts such as "noun" and "verb" were in use in Pāṇini's own time. Pāṇini's own purely formal definition of a word as *sup-tiṅ-antam* "that which ends in a *-sup* suffix or in a *-tiṅ* suffix is a word," and hence as divisible in only two major categories, the noun and the verb, is in any case remarkable. If Pāṇini's definition contrasted with the categories of "noun," "verb," "adverb," "preposition" as we find them in the Nirukta – which is likely but difficult to prove as the relative date of the Nirukta vis-à-vis Pāṇini's work is not established – the latter were apparently relativized by the postulation of the pure technical definition with only two major categories.

According to the fourth thesis, widely accepted by proponents of construction grammar, usage is the basis of the constructions. This is part of a theory on how people learn and use language, and it is the counterpart of theories that place emphasis on inherited components of the language faculty.

Pāṇini is not directly concerned with a theory of individual's language use or language acquisition. As grammarians, however – and not as specialists in psycho-linguistics – the early Pāṇinians such as Kātyāyana and Patañjali clearly base themselves on attested usage which they aim to describe efficiently. It is most likely that we can assume the same for Pāṇini, his contemporary grammarians and his predecessors. There is no trace that it ever was the aim of Pāṇini and early Pāṇinians to describe a mental language capacity.

The contrast between two seventeenth century grammarians in the Pāṇinian tradition will in this respect appear in a different light (cf. Houben 2008a). One among these two, Bhaṭṭoji Dīkṣita, placed Pāṇini and his two early successors, Kātyāyana and Patañjali, on a level of absolute nominal and practical grammatical authority. Although seemingly "saving" the three Pāṇinian *muni*s from distortions by lesser grammarians who come later in the tradition, he in fact cuts himself off from the Pāṇinian "spirit" of usage based grammar. The other, Nārāyaṇa Bhaṭṭa, defended the authority of "non-Pāṇinian" grammarians even if he himself follows Pāṇini's system in great detail and adopted all his central techniques and devices. Although seemingly giving a lower place to Pāṇini it is precisely Nārāyaṇa Bhaṭṭa who preserves the Pāṇinian "spirit" of usage based grammar. Practically all major specialists of Pāṇinian grammar, western and Indian, trace their teacher parentage back to the school of Bhaṭṭoji Dīkiṣita which found its fulfillment in the work of Nāgeśa Bhaṭṭa. Through a configuration of factors, Nārāyaṇa's work was neglected even in his native area (in what is now Kerala), and his distinctive, usage based perspective on Pāṇinian grammar of a sanskrit tradition that can be said to have been "living" at least up to seventeenth century Kerala, has been largely neglected by modern scholars. In the light of the principles of construction grammar it appears worthwhile to review the modern scholars' automatic choice of perspective on Pāṇinian grammar.

3 Computerizing Pāṇini's Grammar

In an important overview of modern Pāṇinian Studies, namely George Cardona's *Recent Research in Pāṇinian Studies* (1999), we read in the concluding section that the author considers the "expanding use of technology in connection with Indology and particularly the application of computer science methods to Pāṇini" a major research direction in Pāṇinian Studies.

Indeed, in the last few decades publications on sanskrit grammar and on sanskrit computational linguistics often express high expectations regarding a "fruitful collaboration between traditional grammarians and engineers" in order to contribute to the solution of "some of the problems of modern technology" (Le Mée 1989: 114, approvingly cited in Cardona 1999: 272). This view[3] harmonizes well with the view on grammar and its purposes dominant in modern linguistics in the past two or three decades: the rules of a grammar should be able "to generate the infinite number of sentences of the language" in such a way that "any speaker, or even a machine, that followed the rules would produce sentences of the language, and if the rules are complete, could produce the potentially infinite number of its sentences" (Searle 2002: 33; cf. Chomsky 1965).

Pāṇini's grammar in which an intricate system of rules occupies a central position has frequently been compared with a computer program. As systematic collections of rules Pāṇini's grammar and a computer program can indeed be compared, but how far can we really take this popular comparison? If the two are so similar, transcribing the rules of Pāṇini's grammar intelligently into an XML-language should yield us a rich computer program describing the sanskrit language. Since at least twenty years there have been ideas to develop "programs replicating Pāṇinian prakriyā" and programs that analyse "strings in terms of Pāṇinian rules" (cp. Cardona 1999 : 272f). In spite of several elaborate and sophisticated attempts in this direction, it seems we are still far from a comprehensive and convincing endresult. Why is it proving so difficult, for at least some twenty years, to computerize Pāṇini's grammar?

Perhaps a major reason is that we are not clear on some crucial issues regarding Pāṇini's grammar. In particular, it remains generally unclear for which aim exactly Pāṇini wrote his grammar and for which aim it was accepted and transmitted by his public. The focus on Pāṇini as an isolated genius has prevented us from rigorously addressing the question: what is the nature of Pāṇini's grammar and what were the aim and context of his grammar in his own time?

According to Cardona (1999: 201), the Aṣṭādhyāyī "presents a synthetic system, whereby affixes are introduced, under meaning and co-occurrence conditions, to verbal and nominal bases, forming syntactic words (*pada*) that bear particular semantic and syntactic relations with each other." Each part in this

[3] While Cardona suggests here he supports the high expectations regarding a "fruitful collaboration between traditional grammarians and engineers," he is elsewhere rightly reticent in accepting detailed parallels between Pāṇini and methods and approaches in modern linguistics.

statement is in itself correct, yet on its own the statement as a whole amounts to a one-sided and incomplete, and in that sense also problematic view of Pāṇini's system. If the system is only synthetic, why would so much attention have been paid to the finished utterances of Vedic texts[4] with all their grammatical exceptions? If the system is synthetic, it must be the abstracted linguistic elements (affixes, verbal and nominal bases) that form the starting point of the synthesis. But then one finds that the system fails entirely in providing guidance to arrive at an acceptable utterance. However, in the practice of modern, early and pre-modern Pāṇinīyas through the ages up to the present, no-one has ever produced a correct form through Pāṇini's system that was not already his starting point, or among his starting options. Usually the correct form is put at the beginning after which it is derived through the system. This is not what modern users of grammar usually do with their grammars, if, for instance, they want to learn a language. We can hence suspect that the aim of Pāṇini's grammar must have been something else. As already indicated, it is therefore useful to see Pāṇini's grammar not as "a device that starts from meaning information" nor to see it as a synthetic system combining affixes and nominal and verbal bases, but as a system that starts with a preliminary statement. The more comprehensive and more realistic view of Pāṇini's grammar as "reconstitutive" rather than one-sidedly "synthetic" gives an important place to unformalized and fundamentally unformalizable domains, which need not be an unsurmountable problem for the designer of a computer program if this is not thought of as a closed system but as a program that interacts with a knowledgeable user who has a starting sentence to be checked.

4 Conclusion and Prospects: The Dhātu-pāṭha as a Central Component of Pāṇini's Grammar

If the fascination with a closed system of rules, which has been more an ideal – not only of modern scholars but also of Pāṇinīyans admired by them such as Bhaṭṭoji and Nāgeśa – than a reality in the case of Pāṇini's grammar, is given up, the interface between the impressive collection of verbal roots together with all the grammatical information it contains, and the collection of rules that are now found together in the Aṣṭādhyāyī can receive more attention. The derivation of a word in a preliminary statement by any potential user of Pāṇini's grammar will normally start with the selection of a root in the Dhātu-pāṭha corresponding to a selected problematic word in his statement. If the grammar user succeeds, he is

---

[4] We may accept, with Bronkhorst 1991: 81-87 and Kelly 1996: 105f (and see now also Bronkhorst 2007), that the process of creating texts coming under Pāṇini's category of *chandas* was probably not yet entirely over in the times of Pāṇini and the Buddha. But compared to the Vedic texts which were ritually employed and transmitted in largely – not yet entirely – fixed forms in Pāṇini's time, linguistic creation in *chandas* must have been marginal, so that the main referent of the term must still be regarded to be "the (established) Vedic texts".

immediately in possesion of crucial grammatical information on this root and is steered on to the rules that can apply. If he does not succeed, he has to go on and search in lists of underived stems, etc. These procedures have little interest from the point of view of generative grammar, but they can be supported by the use of digitital data bases and a consultation program designed by a skilled computer programmer and specialists of Pāṇinian grammar.

The Dhātu-pāṭha has its own problems, for instance the fact that important commentaries on it have not yet been satisfactorily edited. Moreover, in the currently available one associated with Pāṇini's grammar we have not only extensive sections which seem to have predated Pāṇini but also later additions. In general, it seems that new forms have been added over the centuries without discarding outdated ones. Early Dhātu-pāṭhas conserved in Tibetan and the Dhātu-pāṭhas of alternative grammars such as the Sārasvata grammar or the Mugdhabodha, which are still in many respects "Pāṇinian" even if they present themselves as independent, are here of interest not only for the forms they contain but also for those left out. This can help in tracing something of the linguistic reality of two millennia of 'living' sanskrit in India, to which strict followers of Bhaṭṭoji's school have to remain blind.

The challenge of a computerized Pāṇinian grammar together with theoretical incentives derived from construction grammar may thus provide a new impetus to the study of domains and aspects in the work of sanskrit grammarians, and finally of the rich cultural tradition of sanskrit literature, that have been largely neglected till now, among them the domain of the Dhātu-pāṭhas that deserves to be taken up at the point where Liebich and Palsule left it.

References

Bronkhorst, J.: The role of meanings in Pāṇini's grammar. Indian Linguistics 40, 146–157 (1979)

Bronkhorst, J.: Pāṇini and the Veda reconsidered. In: Deshpande, M., Bhate, S. (eds.) Pāṇinian Studies: Professor S.D. Joshi Felicitation Volume, pp. 75–121. Center for South and Southeast Asian Studies, Ann Arbor (1991)

Bronkhorst, J.: Greater Magadha: Studies in the Culture of Early India. E.J. Brill, Leiden (2007)

Brugman, C.M.: The story of over: Polysemy, Semantics, and the Structure of the Lexicon. Garland, New York (1988)

Cardona, G.: Recent Research in Pāinian Studies. Motilal Banarsidass, Delhi (1999)

Chomsky, N.: Aspects of the Theory of Syntax. MIT Press, Cambridge (1965)

Corrigan, R., Eckman, F., Noonan, M.: Linguistic categorization. John Benjamins, Philadelphia (1989)

Croft, W.: Syntactic Categories and Grammatical Relations. Univ. of Chicago Press, Chicago (1991)

Croft, W.: Radical Construction Grammar: Syntactic Theory in Typological Perspective. Oxford Univ. Press, New York (2001)

Croft, W.: Logical and typological arguments for Radical Construction Grammar. In: Fried, M., Östman, J.-O. (eds.) Construction grammar(s): Cognitive and cross-language dimensions. John Benjamins, Amsterdam (2003), http://lings.ln.man.ac.uk/Info/staff/WAC/Papers/RCG-CAL.pdf

Croft, W.: Some implications of Radical Construction Grammar for syntactic organization and semantic structure (in Preparation), http://lings.ln.man.ac.uk/Info/staff/WAC/WACabst.html#inpreprcgimpl

Deane, P.: Limits to attention: A cognitive theory of island phenomena. Cognitive linguistics 2(1), 1–64 (1991)

Fauconnier, G.: Espaces mentaux. Minuit, Paris (1984)

Fauconnier, G.: Mental Spaces. In: Aspects of Meaning Construction in Natural Language. MIT, Cambridge (1985); Cf. Fauconnier 1984. New edn. Cambridge Univ. Press, Cambridge (1994)

Falk, H.: Schrift im alten Indien: Ein Forschungsbericht mit Anmerkungen. Gunter Narr Verlag, Tübingen (1993)

Fillmore, C.J.: Frame semantics. In: Linguistics in the morning calm, pp. 111–138. Hanshin, Seoul (1982)

Fillmore, C.J., Paul, K.: Construction Grammar Coursebook, University of California, Copy Central, Berkeley (1993)

Fillmore, C.J., Kay, P., O'Connor, C.: Regularity and idiomaticity in grammatical constructions: The case of let alone. Language 64, 501–538 (1988)

Givón, T.: Prototypes: Between Plato and Wittgenstein. In: Craig, C. (ed.) Noun classes and categorization: proceedings of a symposium on categorization and noun classification. John Benjamins, Philadelphia (1986)

Goldberg, A.E.: Constructions. In: A Construction Grammar Approach to Argument Structure. Chicago Univ. Press, Chicago (1995)

Goldberg, A.E.: Jackendoff and construction-based grammar. Cognitive Linguistics 7(1), 3–19 (1996)

Haiman, J. (ed.): Natural Syntax: Iconicity and Erosion. Cambridge Univ. Press, Cambridge (1985)

von Hinüber, O.: Der Beginn der Schrift und frühe Schriftlichkeit in Indien. Franz Steiner, Wiesbaden (1989)

Hopper, P.J., Thompson, S.A.: The discourse basis for lexical categories in Universal Grammar. Language 60, 703–752 (1984)

Houben, J.E.M.: The Sanskrit tradition. In: van Bekkum, W., Houben, J., Sluiter, I., Versteegh, K. (eds.) The Emergence of Semantics in Four Linguistic Traditions: Hebrew, Sanskrit, Greek, Arabic, pp. 49–145. John Benjamins, Amsterdam (1997)

Houben, J.E.M.: Meaning Statements' in Pāṇini's grammar: on the purpose and context of the Aṣṭādhyāyī. Studien zur Indologie und Iranistik 22(1999 [2001]), 23–54 (1999)

Houben, J.E.M.: Three Myths in Modern Pāṇinian Studies (Review article of George Cardona, Recent Research in Pāinian Studies, Delhi 1999.) Asiatische Studien/Études Asiatiques 57(1), 121–179 (2003)

Houben, J.E.M.: Sur la théorie du nombre de Bhartṛhari (Review article of Pascale Haag, Le Saṁkhyāsamuddeśa du Bhartṛhari (théorie du nombre), Paris 2005.) Histoire – Epistémologie – Language, Tome XXVIII, 157–166 (2006)

Houben, J.E.M.: Bhaṭṭoji Dīkṣita's 'small step' for a Grammarian and 'Giant Leap' for Sanskrit Grammar. Journal of Indian Philosophy 36, 563–574 (2008a)

Houben, J.E.M.: Bhartṛhari as a ‘cognitive linguist’. In: Chaturvedi, M. (ed.) Proceedings of the International Seminar on Bhartṛhari, December 12-14, 2003. Motilal Banarsidass, New Delhi (2008b)

Joshi, S.D., Roodbergen, J.A.F.: Patañjali’s Vyākaraña-Mahābhāṣya, Kārakāhnika. Introduction, translation and notes, pp. 1.4.23–1.4.55. Centre of Advanced Study in Sanskrit, University of Poona, Poona (1975)

Kaldewaij, J.: Structuralisme en Transformationeel Generatieve Grammatica. Dissertation. Utrecht University (1986)

Kelly, J.D.: What was Sanskrit for? Metadiscursive strategies in ancient India. In: Houben, J. (ed.) Ideology and Status of Sanskrit: Contributions to the History of the Sanskrit Language, pp. 87–107. E.J. Brill, Leiden (1996)

Kiparsky, P.: Some theoretical problems in Pāṇini’s grammar. Post-graduate and Research Department Series No. 16. Bhandarkar Oriental Research Institute, Pune (1982)

Kiparsky, P.: On the Architecture of Pāṇini’s Grammar. In: Three lectures delivered at the Hyderabad Conference on the Architecture of Grammar and at the University of California, L.A. (2002), downloaded on December 15, 2006 `http://www.stanford.edu/~Papershyderabad.pdf`

Kiparsky, P., Stall, F.: Syntactic and semantic relations in Pāṇini. Foundations of Language 5, 83–117 (1969)

Lakoff, G.: Linguistic gestalts. Chicago Linguistic Society 13, 225–235 (1977)

Lakoff, G.: Women, Fire and Dangerous Things. Chicago Univ. Press, Chicago (1987)

Langacker, R.W.: Observations and speculations on subjectivity. In: Haiman, J. (ed.) Natural Syntax: Iconicity and Erosion, pp. 109–150. Cambridge Univ. Press, Cambridge (1985)

Langacker, R.W.: Foundations of Cognitive Grammar 1. Stanford Univ. Press, Stanford (1987)

Langacker, R.W.: A usage-based model. In: Topics in Cognitive Linguistics (by Rudzka-Ostyn, Brygida), pp. 127–161 (1988)

Langacker, R.W.: A dynamic usage-based model. In: Barlow, M., Kemmer, S. (eds.) Usage-based Models of Language, pp. 1–63. CSLI Publications, Stanford (2000)

Le Mée, J.: Pāṇinīyas and engineers. In: Kumar, A., et al. (eds.) Studies in Indology: Prof. Rasik Vihari Joshi Felicitation Volume, pp. 113–121. Shree Publishing House, New Delhi (1989)

Liebich, B.: Zur Einführung in die indische einheimische Sprachwissenschaft, II: Historische Einführung und Dhātupāṭha. C. Winter, Heidelberg (1919)

Palsule, G.B.: The Sanskrit Dhātupāṭhas: A Critical Study. University of Poona, Poona (1961)

Rosch, E.: Natural Categories. Cognitive Psychology 4, 328–350 (1973)

Rosch, E., et al.: Basic objects in natural categories. Cognitive Psychology 8, 382–439 (1976)

Searle, J.R.: End of the Revolution. The New York Review of Books, February 28, p. 33 (2002)

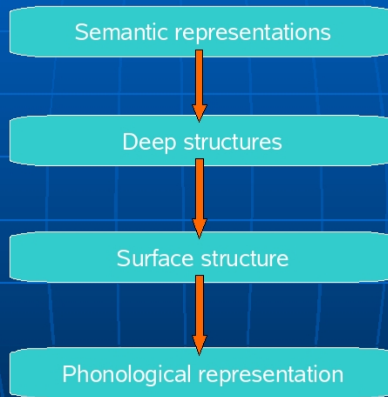
Staal, F.: Universals: Studies in Indian Logic and Linguistics. Univ. of Chicago Press, Chicago (1988)

Talmy, L.: The relation of grammar to cognition. In: Waltz, D. (ed.) Theoretical Issues in Natural Language Processing, vol. 2, Coordinated Science Laboratory, Univ. of Illinois, Champaign (1978)

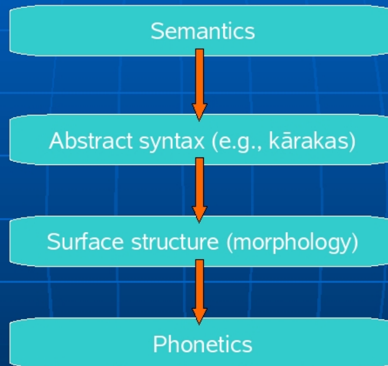
Talmy, L.: Force dynamics in language and thought. In: Eilfort, W., Kroeber, P., Peterson, K. (eds.) CLS, Parasession on Causatives and Agentivity, pp. 293–337 (1985)

Taylor, J.R.: Linguistic categorization: prototypes in linguistic theory. Clarendon Press, Oxford (1989)
Wierzbicka, A.: The semantics of 'internal dative' in English. Quaderni di Semantica 7, 155–165 (1986)
Wierzbicka, A.: The Semantics of Grammar. John Benjamins, Philadelphia (1988)

Appendix

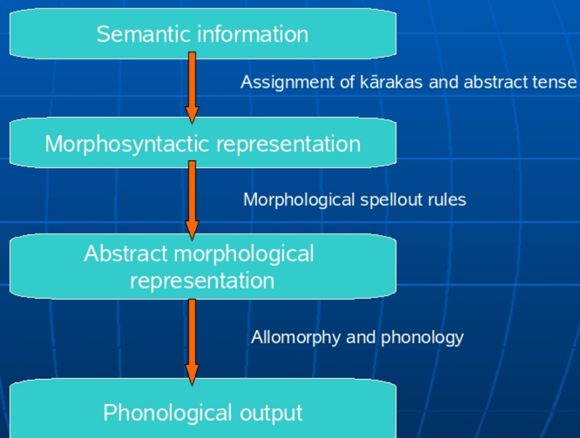


Kiparsky 1982

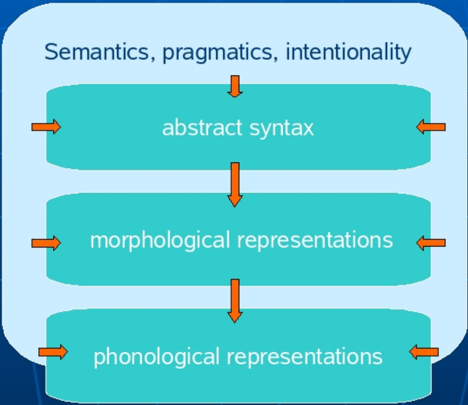


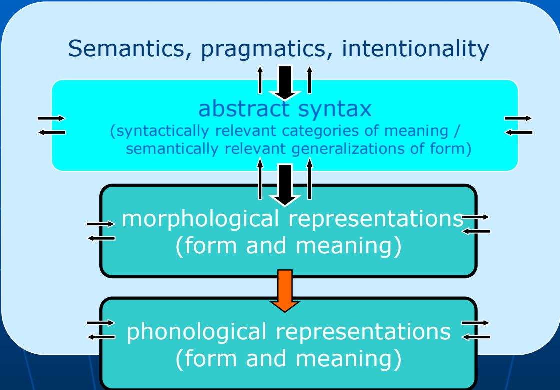
Construction grammar J.E.M. Houben 2

Kiparsky 2002 “Architecture … ”



Construction grammar J.E.M. Houben 3


Houben 1999 "Meaning statements … "
Semantics, pragmatics, intentionality
abstract syntax
morphological representations
phonological representations
Construction grammar J.E.M. Houben
4


Form and meaning,
derivation ( ), consultation ( ), labeling ( )
Semantics, pragmatics, intentionality
abstract syntax
(syntactically relevant categories of meaning / semantically relevant generalizations of form)
morphological representations
(form and meaning)
phonological representations
(form and meaning)
Construction grammar J.E.M. Houben
1

Annotating Sanskrit Texts Based on Śābdabodha Systems

K.V. Ramkrishnamacharyulu

Rastriya Sanskrit Vidyapeetham, Tirupati, India

1 वन्दनम्

श्री रामो विजयतेतराम्। श्री रामाय नमः। श्री गुरुभ्यो नमः। श्रीमद्रामानुजाय नमः।

2 Introduction

Though Sanskrit has a huge repository of texts as well as well discussed grammar formalism, we still neither have a full fledged parser for Sanskrit based on this formalism nor do we have any annotated text.

We propose here a tagging scheme for manual tagging of Sanskrit texts, based on different grammatical relations that have been discussed by different schools of śābdabodha systems. We hope the tagging scheme proposed here serves as a good starting point for manual annotation.

The process of Śābdabodha involves identifying the relations between different words in a sentence. The traditional model of sentence analysis which is mainly based on Pāṇinian model provides us with various relations that are mainly Syntactico-Semantic in nature. Tradition treats a sentence as a series of modifier-modified relations(M.M.R.). *Ākāṅkṣā* (expectancy) plays a major role in establishing these relations.

Sentence is a group of words that are bound together by ākāṅkṣā. Sentences are of two types: vākya (simple sentence) and mahā-vākya (complex sentence).

Vākya is a group of words with one verb. Other words in the sentence satisfy the ākāṅkṣās (expectancies) of the main verb. These ākāṅkṣās are called utthita ākāṅkṣā(Natural expectancies). The main verb is the mukhya viśeṣya (head) which is modified by other words (modifiers) in the sentence.

Mahā vākya is a group of sentences which are inter-related and denote a single meaning. Here the head of the main sentence is modified by the heads of the other (subordinate) sentences. As such the relations between the main verb of the main sentence and the main verbs in sub-ordinate sentences denote the inter-sentential relations. The ākāṅkṣā between the activities i.e. heads of the sentences is not natural but are triggered by special words in the sentence construction. These ākāṅkṣās, therefore, are called *utthāpyākāṅkṣā*.

The relations, thus, are of two types: external relations - relations between sentences and internal relations - relations between words within the sentences.

In what follows, we list different inter-sentential and intra-sentential relations. We indicate, wherever possible, the marker which helps in identifying these

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 26–39, 2009.

relations, semantics associated with these relations, language conventions associated with them, if any, followed by the proposed tag[1] with an example sentence.

3 Proposed Tagset

3.1 Inter-sentential Relations

These relations are denoted by certain link words and non-finite verb forms. The ākāṅkṣā between two sentences is known only if the link words are present in the sentence. All such relations are marked by (...)_R where (...) denotes the part of the sentence, R is the inter-sentential relation. The inter-sentential relations may be further subclassified as

1. Relations denoted by non-finite verbs
2. Relations denoted by certain link words

Relations denoted by Non-finite verbs

1. पूर्वकालीनत्वम् (pūrvakālīnatvam)
   liṅgam: ktvā ending non-finite verb form
   Meaning: ktvā- ending verb form denotes the activity which preceeds an activity denoted by the main verb.
   Properties: The kartā of the ktvā ending verb is shared with that of the main verb[2].
   Proposed tag: (...)_ktvā
   Example: (रामः दुग्धं पीत्वा)_ktvā शालां गच्छति) ।
   (rāmaḥ (dugdhaṁ pītvā)_ktvā śālāṃ gacchati).

2. प्रयोजनम् (purpose of the main activity)
   liṅgam: tumun-ending nonfinite verb form
   Meaning: The tumun-ending verb denotes the purpose of the main activity
   Properties: The kartā is shared in some cases. In some cases the karma or sampradāna of the main activity becomes the kartā of the purpose activity.
   proposed tag: (...)_tumun
   Example: (अहं प्रतिदिनं (योगशास्त्रं पठितुं)_tumun विद्यालयं गच्छामि)।
   (ahaṃ pratidinaṃ (yogaśāstraṃ paṭhituṁ)_tumun vidyālayaṃ gacchāmi).
   (अहं भवन्तं मम गृहे (भोक्तुम्)_tumun आह्वयामि) ।
   (ahaṃ bhavantaṃ mama gṛhe (bhoktum)_tumun āhvayāmi).

[1] Though technically name of the relation should end in the bhāva pratyaya 'tva' as in 'kartṛtva', 'karmatva' etc., since we tag the words, we tag them as 'kartā', 'karma', etc.
Thus in the sentence
रामः गृहं गच्छति। (rāmaḥ gṛhaṁ gacchati),
rāma has kartṛtva of the kriyā denoted by *gam*, but we tag it as
रामः_kartā गृहं गच्छति। (rāmaḥ_kartā gṛhaṁ gacchati).

[2] समानकर्तृकयोः पूर्वकाले 3.4.21, (samānakartṛkayoḥ pūrvakāle).

3. समकालीनत्वम् (Simultaneity)
   liṅgam: śatṛ or śānac
   Meaning: It denotes the activity occuring simultaneously with the main verb
   Properties: The kartā is shared with the main activity
   proposed tag: (...)_śatṛ/(...)_śānac
   Example: (बालकः (जलं पिबन्)_śatṛ गच्छति) ।
   (bālakaḥ (jalaṃ piban)_śatṛ gacchati).
   (बालकः (शयानः)_śānac हसति) ।
   (bālakaḥ (śayānaḥ)_śānac hasati).

4. भावलक्षणसप्तमी (bhāvalakṣāṇa saptamī)

   – अनन्तरकालिकत्वम् (Time of the completion of preceeding activity)
     liṅgam: kta - ending in 7^{th} case
     proposed tag: (...)_kta_7
     Example: ((रामे वनं गते)_kta_7 दशरथः खिन्नः) ।
     ((rāme vanaṃ gate)_kta_7 daśarathaḥ khinnaḥ).

   – समकालीनत्वम् (simultaneous events)
     liṅgam: śatṛ or śānac - ending in 7^{th} case
     propsed tag: (...)_śatṛ$_7$
     Example: ((रामे वनं गच्छति)_śatṛ$_7$ सीता अनुसरति) ।
     (rāme vanaṁ gacchati)_śatṛ$_7$ sītā anusarati.

   – पूर्वकालीनत्वम् (time of the main activity before the starting of the subordinate activity)
     liṅgam: lṛṭ + śatṛ or śānac ending in 7^{th} case
     Proposed tag: (...)_lṛṭ-śatṛ$_7$
     Example: ((गोषु धोक्ष्यमाणासु)_lṛṭ-śatṛ$_7$ गतः) ।
     ((goṣu dhokṣyamāṇāsu)_lṛṭ-śatṛ$_7$ gataḥ).

Relations denoted by words

1. समानकालीनत्वम् (samānakālīnatvam)

   – liṅgam: yadā - tadā or yasmin kāle - tasmin kāle
     yadā or yasmin kāle in the subordinate sentence typically in the beginning and tadā or tasmin kāle in the beginning of the main sentence
     proposed tag: (...)_yadā$_1$
     Example: (तदा मयूरः नृत्यति (यदा मेघः वर्षति)_yadā$_1$) ।
     (tadā mayūraḥ nṛtyati (yadā meghaḥ varṣati)_yadā$_1$)

   – liṅgam: yadā or yasmin kāle
     Only yadā or yasmin kāle is present, and tadā or tasmin kāle is absent.
     proposed tag: (...)_yadā$_2$

Example: (मयूरः नृत्यति (यदा मेघः वर्षति)_yadā$_2$) ।
(mayūraḥ nṛtyati (yadā meghaḥ varṣati)_yadā$_2$)

– liṅgam: tadā or tasmin kāle
Only tadā or tasmin kāle is present, and yadā or yasmin kāle is absent.
proposed tag: (...)_tadā
Example: (मेघः वर्षति (तदा मयूरः नृत्यति)_tadā) ।
(meghaḥ varṣati (tadā mayūraḥ nṛtyati)_tadā)

2. प्रतिबन्धः (conditional relation)

– liṅgam: yadi - tarhi
'yadi' in the beginning of a subordinate sentence and 'tarhi' in the main sentence;
proposed tag: (...)_yadi$_1$
Example: ((यदि त्वम् इच्छसि)_yadi$_1$ तर्हि अहं भवतः गृहम् आगच्छामि) ।
((yadi tvam icchasi)_yadi$_1$ tarhi ahaṁ bhavataḥ gṛham āgacchāmi).

– liṅgam: yadi
Only yadi is used,
proposed tag: (...)_yadi$_2$
Example: (अहम् आगमिष्यामि (यदि भवान् अपेक्षितं सौलभ्यं विधास्यति)_yadi$_2$) ।
(aham āgamiṣyāmi (yadi bhavān apekṣitaṁ saulabhyaṁ vidhāsyati)_yadi$_2$).

– liṅgam: tarhi
Only 'tarhi' is used, and the word 'yadi' is missing. proposed tag (...)_tarhi
Example: (त्वम् इच्छसि (तर्हि अहं भवतः गृहम् आगच्छामि)_tarhi) ।
(tvam icchasi (tarhi ahaṁ bhavataḥ gṛham āgacchāmi)_tarhi).

– liṅgam: cet
presence of the word 'cet' proposed tag: (...)_cet
Example: ((त्वम् इच्छसि चेत्)_cet अहं भवतः गृहम् आगच्छामि) ।
((tvam icchasi cet)_cet ahaṁ bhavataḥ gṛham āgacchāmi).

– liṅgam: tarhi eva
group the words from the beginning up to tarhi_eva as one sentence, and the rest as second sentence
proposed tag: (...)_tarhi_eva
Example: ((त्वम् इच्छसि तर्हि एव)_tarhi_eva अहं भवतः गृहम् आगच्छामि) ।
((tvam icchasi tarhi eva)_tarhi_eva ahaṁ bhavataḥ gṛham āgacchāmi)).

3. कारणसत्वेऽपि कार्याभावः, कारणाभावेऽपि कार्योत्पत्तिः (Non productive effort (or) product without cause)
– liṅgam: yadyapi – tathāpi
proposed tag (...)_yadyapi$_1$

Example: ((यद्यपि अयं बहु प्रयासं कृतवान्)_yadyapi$_1$ तथापि परीक्षा तु अनुत्तीर्णा) ।
((yadyapi ayaṁ bahu prayāsaṁ kṛtavān)_yadyapi$_1$ tathāpi parīkṣā tu anuttīrṇā)

– liṅgam: yadyapi
Example: ((यद्यपि अनेन बहु प्रयासः कृतः)_yadyapi$_2$ परीक्षा तु अनुत्तीर्णा) ।
((yadyapi anena bahu prayāsaḥ kṛtaḥ)_yadyapi$_2$ parīkṣā tu anuttīrṇā).

– liṅgam: tathāpi proposed tag (...)_tathāpi
Example: (अयं तथा न कुशलः (तथापि प्रथमपुरस्कारं लब्धवान्)_tathāpi) ।
(ayaṁ tathā na kuśalaḥ (tathāpi prathamapuraskāraṁ labdhavān)_tathāpi)

– liṅgam: athāpi or evamapi
proposed tag (...)_athāpi
Example: (परीक्षायाम् अहम् अनुत्तीर्णः (अथापि पुनः लिखिष्ये)_athāpi) ।
parīkṣāyām aham anuttīrṇḥ (athāpi punaḥ likhiṣye)_athāpi)

4. हेतुहेतुमद्भावः (cause and effect)

– liṅgam: yataḥ-tataḥ or yasmāt-tasmāt
proposed tag (...)_yataḥ$_1$
Example: ((यतः अयं समये नागतः)_yataḥ1 ततः प्रवेशपरीक्षायां नानुमतः) ।
((yataḥ ayaṁ samaye nāgataḥ)_yataḥ1 tataḥ praveśaparīkṣāyāṃ nānumataḥ)

– liṅgam: yataḥ or yasmāt
proposed tag (...)_yataḥ$_2$
Example: (प्रवेशपरीक्षायां नानुमतः अयं (यतः समये नागतः)_yataḥ$_2$) ।
(praveṣaparīkṣāyāṁ nānumataḥ ayaṃ (yataḥ samaye nāgataḥ)_yataḥ$_2$)

– liṅgam: tataḥ or tasmāt or ataḥ
proposed tag (...)_tataḥ
Example: (अयं समये नागतः (ततः अयं परीक्षायां नानुमतः) _tataḥ)।
(ayaṃ samaye nāgataḥ tataḥ ayaṃ parīkṣāyāṃ nānumataḥ)

5. अनन्तरकालीनत्वम् (following action)
liṅgam: tataḥ or tatastataḥ or anantaram or atha
proposed tag: (...)_atha
Example: (प्रथमम् अहं शृणोमि (अथ लिखामि)_atha) ।
(prathamam ahaṁ śṛṇomi (atha likhāmi)_atha)

6. समुच्चयः (conjunction)
liṅgam: api_ca or kiṃ_ca
proposed tag (...)_apica
Example: (भिक्षाम् अट (अपिच गामानय)_apica) ।
(bhikṣām aṭa (apica gāmānaya)_apica)

7. समानाधिकरणत्वम् (co-location)
   - liṅgam: yatra - tatra or yasmin - tasmin
     proposed tag: (...)_yatra$_1$
     Example: ((यत्र नार्यस्तु पूज्यन्ते)_yatra$_1$ रमन्ते तत्र देवताः) ।
     ((yatra nāryastu pūjyante)_yatra$_1$ ramante tatra devatāḥ)

   - liṅgam: yatra or yasmin
     proposed tag: (...)_yatra$_2$
     (अहो बृन्दावनं रम्यं (यत्र गोवर्धनो गिरिः)_yatra$_2$) ।
     (aho bṛndāvanaṃ ramyaṃ (yatra govardhano giriḥ)_yatra$_2$)

   - liṅgam: tatra or tasmin
     proposed tag: (...)_tatra
     Example: ((तत्र स्नात्वा नरो राजन्)_tatra गोसहस्रफलं लभेत) ।
     ((tatra snātvā naro rājan)_tatra gosahasraphalaṃ labheta)

8. असाफल्यम् (non-fulfilment of expected activity)
   liṅgam: kintu or parantu
   proposed tag (...)_kintu
   Example: (गजेन्द्रः तीव्रप्रयत्नम् अकरोत् (किन्तु नक्रग्रहात् न मुक्तः)_kintu) ।
   (gajendraḥ tīvraprayatnam akarot (kintu nakragrahāt na muktaḥ)_kintu)

4 Sentence Internal Relations

These are of two types

- related to the words denoting activity,
- related to other words

4.1 Relations Related to the Activity-Denoting Words

These relations are triggered by the vibhaktis. However one vibhakti may indicate several relations. It is the context which indicates a particular relation. we mark these relations by REL where REL stands for the relation label.
These relations are also of two types:

- kāraka relations
- non-kāraka relations

- kāraka relations
  - kartā k_1
  - karma k_2
  - karaṇa k_3
  - sampradāna k_4
  - apādāna k_5
  - adhikaraṇa k_7

These kāraka relations may also be further classified as

- **कर्ता (kartā)** (k_1)

  * default (k_1)
    देवदत्तः_k_1 पचति।
    devadattaḥ_k_1 pacati
    रथः_k_1 गच्छति।
    rathaḥ_k_1 gacchati

  * अनुभवी (experiencer(k_1-e)
    Example: घटो_k_1-e नश्यति।
    ghaṭo_k_1-e naśyati.
    पुत्रः_k_1-e जायते।
    putraḥ_k_1-e jāyate.
    सः_k_1-e सुखम् अनुभवति।
    saḥ_k_1-e sukham anubhavati.

  * अमूर्तः (abstract) (k_1-a)
    Example: क्रोधः_k_1-a आगच्छति।
    krodhaḥ_k_1-a āgacchati.

  * प्रयोजकः (prayojakaḥ) (k_1-p)
    देवदत्तः_k_1-p विष्णुमित्रेण पाचयति।
    (devadattaḥ_k_1-p viṣṇumitreṇa pācayati.

  * प्रयोज्यः (prayojyaḥ) (k_1-j)
    देवदत्तः विष्णुमित्रेण_k_1-j पाचयति।
    devadattaḥ viṣṇumitreṇa_k_1-j pācayati.

  * मध्यस्थः (madhyasthaḥ) (k_1-m)
    देवदत्तः यज्ञदत्तेन_k_1-m विष्णुमित्रेण पाचयति।
    devadattaḥ (yajñadattena)_k_1-m viṣṇumitreṇa pācayati.

  * अभिप्रेरकः/उत्प्रेरकः (cause for temptation) (k_1-t)
    Example: मोदकः_k_1-t रोचते।
    modakaḥ_k_1-t rocate.

  * कर्म-कर्तृ (karma-kartṛ) (k_1-k_2)
    Example: भिद्यते काष्ठः_k_1-k_2 स्वयमेव।
    bhidyate kāṣthaḥ_k_1-k_2 svayameva.
    पच्यते ओदनः_k_1-k_2 स्वयमेव।
    pacyate odanaḥ_k_1-k_2 svayameva.

  * करण-कर्तृ (karaṇa-kartṛ) (k_1-k_3)
    Example: असिः_k_1-k_3 छिनत्ति।
    asiḥ_k_1-k_3 chinatti.

  * षष्ठी - कर्ता (ṣaṣṭhī kartā) (K1-6)
    Example: आचार्यस्य_k_1-6 अनुशासनम्।
    ācāryasya_k_1-6 anuśāsanam.

- **कर्म (karma)** (k_2)
  * default (k_2)
    Example: शत्रून्_k_2 जयति।
    śatrūn_k_2 jayati.
    ओदनं_k_2 भुङ्क्ते।
    odanaṃ_k_2 bhuṅkte.

  * उत्पाद्यम् (created)(k_2-u)
    Example: ओदनं_k_2-u पचति।
    odanaṃ_k_2-u pacati.

  * विकार्यम् (raw meterial)(k_2-v)
    Example: सुवर्णं_k_2-v कुण्डलं करोति।
    suvarṇaṃ_k_2-v kuṇḍalaṃ karoti.

  * प्रयोज्य-कर्ता (prayojya-kartā) (k_2-j)
    Example: बालं_k_2-j क्षीरं पाययति।
    bālaṃ_k_2-j kṣīraṃ pāyayati.

  * आधारः (location) (k_2-l)
    Example: वैकुण्ठम्_k_2-l अधिशेते।
    vaikuṇṭham_k_2-l adhiśete.

  * देशः (village, town, state, country etc) (k_2-p)
    Example: कुरून्_k_2-p स्वपिति।
    kurūn_k_2-p svapiti.

  * कालः (time) (k_2-t)
    Example: मासम्_k_2-t आस्ते।
    māsam_k_2-t āste.

  * भावः (activity) (k_2-a)
    Example: गोदोहम्_k_2-a आस्ते।
    godoham_k_2-a āste.

  * मार्गः (road measurment) (k_2-m)
    Example: क्रोशम्_k_2-m आस्ते।
    krośam_k_2-m āste.

  * सम्प्रदानम् (recipient) (k_2-k_4)
    Example: पशुना रुद्रं_k_2-k_4 यजते।
    paśunā rudraṃ_k_2-k_4 yajate.

* अनीप्सितम् (not intended) (k_2-an)
  Example: ग्रामं गच्छन् तृणं_k_2-an स्पृशति।
  grāmaṃ gacchan tṛṇaṃ_k_2-an spṛśati.

* अकथितम् (not expected) (k_2-un)
  Example: गोपः गां_k_2-un दोग्धि पयः।
  gopaḥ gāṃ_k_2-un dogdhi payaḥ

* गति-कर्म (gati-karma) (k_2-g)
  Example: रामः ग्रामं_k_2-g गच्छति।
  rāmaḥ grāmaṃ_k_2-g gacchati.

* करणम् (instruments of playing) (k_2-k_3)
  Example: अक्षान्_k_2-k_3 दीव्यति।
  akṣān_k_2-k_3 dīvyati.
  कन्दुकं_k_2-k_3 क्रीडति।
  kandukaṃ_k_2-k_3 krīḍati.

* यं प्रति कोपः (yaṃ prati kopaḥ) (k_2-k)
  Example: क्रूरम्_k_2-k अभिक्रुध्यति।
  krūram_k_2-k abhikrudhyati.

* मन्य-कर्म (in disrespect) (k_2-d)
  Example: न त्वां तृणाय_k_2-d / तृणं_k_2-d मन्ये।
  na tvāṃ tṛṇāya_k_2-d / tṛṇaṃ_k_2-d manye.

* षष्ठी-कर्म (ṣaṣṭhī-karma) (k_2-6)
  Example: शब्दानाम्_k_2-6 अनुशासनम्।
  śabdānām_k_2-6 anuśāsanam.

- करणम् (instrument) (k_3)

  * default (k_3)
    बालः कुञ्चिकया तालम् उद्घाटयति।
    bālaḥ kuñcikayā tālam udghāṭayati.

  * कर्म (karma) (k_3-k_2)
    Example: पशुना_k_3-k_2 रुद्रं यजते।
    paśunā_k_3-k_2 rudraṃ yajate.

  * परिक्रयणम् (money in bonded labour) (k_3-m)
    Example: शतेन_k_3-m परिक्रीणाति।
    śatena_k_3-m parikrīṇāti.

- सम्प्रदानम् (recipient) (k_4)
  - सत्वाश्रयः (recipient with ownership) (k_4-o)
    देवदत्तः ब्राह्मणाय_k_4-o गां ददाति।
    devadattaḥ brāhmaṇāya_k_4-o gāṃ dadāti.
  - स्वीकर्ता (recipient without ownership) (k_4)
    देवदत्तः रजकाय_k_4-o वस्त्रं प्रक्षालनाय ददाति।
    devadattaḥ rajakāya_k_4-o vastraṃ prakṣālanāya dadāti.
  - क्रियया अभिप्रेतः (intended to relate with activity) (k_4-i)
    example: पत्ये_k_4-i शेते।
    patye_k_4-i śete.
  - ज्ञीप्स्यमानः (addressed through praise etc.) (k_4-a)
    example: कृष्णाय_k_4-a श्लाघते।
    kṛṣṇāya_k_4-a ślāghate.
  - उत्तमर्णः (a creditor) (k_4-u)
    example: देवदत्ताय_k_4-u शतं धारयति।
    devadattāya_k_4-u śataṃ dhārayati.
  - ईप्सितम् (desired) (k_4-d)
    Example: पुष्पेभ्यः_k_4-d स्पृहयति।
    puṣpebhyaḥ_k_4-d spṛhayati.
  - यं प्रति कोपः सः (point of anger) (k_4-k)
    Example: हरये_k_4-k क्रुध्यति।
    haraye_k_4-k krudhyati.
  - प्रीयमाणः (location of desire) (k_4-p)
    Example: देवदत्ताय_k_4-p रोचते मोदकः।
    devadattāya_k_4-p rocate modakaḥ
  - यस्य विप्रश्नः (enquiry about) (k_4-e)
    Example:- कृष्णाय_k_4-e राध्यति।
    kṛṣṇāya_k_4-e rādhyati.
  - परिक्रयणम् (money in bonded labour) (k_4-b)
    Example: देवदत्तः शताय_k_4-b परिक्रीतः।
    devadattaḥ śatāya_k_4-b parikrītaḥ
- अपादानम् (apādānam) (k_5)
  - default (point of departure/seperation) (k_5)
    वृक्षात्_k_5 पर्णं पतति।
    vṛkṣāt_k_5 parṇaṃ patati.

* भय-हेतुः (cause of fear) (k_5-f)
  Example: गृहस्थः चोरात्_k_5-f बिभेति।
  gṛhasthaḥ corāt_k_5-f bibheti.

* आख्यात-उपयोगे (teacher) (k_5-u)
  Example: छात्रः उपाध्यायात्_k_5-u अधीते।
  chātraḥ upādhyāyāt_k_5-u adhīte.

* यस्मात् वारणम्(point for obstruction) (k_5-o)
  Example: कूपात्_k_5-o अन्धं वारयति।
  kūpāt_k_5-o andhaṃ vārayati.

* यस्य / यस्या अदर्शनम् इष्टं सः / सा (person intended not to be seen) (k_5-n)
  Example: मातुः_k_5-n निलीयते कृष्णः।
  mātuḥ_k_5-n nilīyate kṛṣṇaḥ

* प्रकृतिः (raw material) (k_5-p)
  Example: मृदः_k_5-p घटः जायते।
  mṛdaḥ_k_5-p ghaṭaḥ jāyate.

* प्रभवः (place of first appearence) (k_5-a)
  Example: हिमवतः_k_5-a गङ्गा प्रभवति।
  himavataḥ_k_5-a gaṅgā prabhavati.

* पराजयः (defeat from activity)[3] (k_5-d)
  Example: अध्ययनात्_k_5-d पराजयते।
  adhyayanāt_k_5-d parājayate.

- अधिकरणम् (location) (k_7)
  * कालः (time) (k_7-t)
    Example: त्रेतायुगे_k_7-t रामः आसीत्।
    tretāyuge_k_7-t rāmaḥ āsīt.

  * देशः deśaḥ (place) (k_7-p)
    Example: रामः आयोध्यायाम्_k_7-p आसीत्।
    rāmaḥ āyodhyāyām_k_7-p āsīt.

  * विषयः viṣayaḥ (other than above) (k_7-v)
    Example: मोक्षे_k_7-v इच्छा अस्ति।
    mokṣe_k_7-v icchā asti.

  * समयस्य अवधिः (time duration) (k_7-td)
    Example: जनवरीतः_k_5-a (मई पर्यन्तं)_k_7-td विरामः।
    janavarītaḥ_k_5-a (maī paryantaṁ)_k_7-td virāmaḥ.

[3] पराजेः असोढः parājeḥ asoḍhaḥ 1.4.26.

  - ∗ अन्तराल-देशः (place in between) (k_7-pd)
    Example: तिरुपतितः_k_5-a चन्द्रगिरिपर्यन्तं_k_7-td भवनानि सन्ति।
    tirupatitaḥ_k_5-a candragiriparyantaṁ_k_7-td bhavanāni santi.

– अकारकसम्बन्धः साक्षात् क्रियया (Non kāraka relations, but direct relations with the activity)

- सम्बोधनम् (addressed) (radr)
  Example: भो राम_radr माम् उद्धर।
  bho rāma_radr mām uddhara.

- प्रसज्यप्रतिषेधः (uncompounded negation) (rneg)
  Example: रामः गृहं न_rneg गच्छति।
  rāmaḥ gṛhaṃ na_rneg gacchati.

- साम्यम् (similarity) (rs)
  Example: ब्राह्मणवत्_rs अधीते।
  brāhmaṇavat_rs adhīte.

- क्रिया - आवृत्त्यन्तरालसमयः (time duration between the repetition of the same activity) (rtd)
  Example: अद्य भुक्त्वा दिनद्वयात्_rtd भोक्ता।
  adya bhuktvā dinadvayāt_rtd bhoktā.

- तादर्थ्य (purpose) (rtv)
  Example: छात्रः अध्ययनाय_rtv विद्यालये वसति।
  chātraḥ adhyayanāya_rtv vidyālaye vasati.
  सा क्रयणाय_rtv आपणं गच्छति
  sā krayaṇāya_rtv āpaṇaṃ gacchati.

- हेतुः (cause) (rhv)
  Example: विद्यार्थी अध्ययनेन_rhv विद्यालये वसति।
  vidyārthī adhyayanena_rhv vidyālaye vasati.

- वीप्सा (repitition) (rrpt)
  Example: शकुन्तला आश्रमे प्रतिवृक्षं_rrpt सिञ्चति।
  śakuntalā āśrame prativṛkṣaṃ_rrpt siñcati.

- क्रिया-आवृत्ति-गणना (counting of repetition) (rcrpt)
  Example: बालकः पाठं पञ्चवारं_rcrpt पठति।
  bālakaḥ pāṭhaṁ pañcavāraṁ_rcrpt paṭhati.

- क्रियाविशेषणम् (manner adverb) (rad)
  Example: हस्ती मार्गे मन्दं_rad गच्छति।
  hastī mārge mandaṃ_rad gacchati.
  मृगः वेगेन_rad धावति।
  mṛgaḥ (vegena)_rad dhāvati.

- अत्यन्त-सम्बद्धः कालः (complete relation with time) (rt2)
  Example: बालकः गुरुकुले मासम्_rt2 अधीतः ।
  bālakaḥ gurukule māsam_rt2 adhītaḥ

- अत्यन्त-सम्बद्धः मार्गः (complete relation with road) (rr2)
  Example: पाठः क्रोशम्_rr2 अधीतः ।
  pāṭhaḥ krośam_rr2 adhītaḥ

- अत्यन्त-सम्बद्धः कालः (प्रयोजनः) सफलः (complete relation with time with result) (rt3)
  Example: बालकेन मासेन_rt3 अनुवाकः अधीतः ।
  bālakena māsena_rt3 anuvākaḥ adhītaḥ.

- अत्यन्त-सम्बद्धः मार्गः (प्रयोजनः) सफलः (complete relation with road with result) (rr3)
  बालकेन क्रोशेन_rr3 अनुवाकः अधीतः ।
  bālakena krośena_rr3 anuvākaḥ adhītaḥ.

– Other Relations
  - षष्ठी (ṣaṣṭhī relation) (r6)
    (अध्यापकस्य)_r6 पुस्तकं छात्राः पठन्ति ।
    adhyāpakasya_r6 pustakaṃ chātrāḥ paṭhanti.

  - आरम्भसमयः मापने (starting point of time) (rst5)
    Example: कार्तिक्याः_rst5 आग्रहायणी मासे ।
    kārtikyāḥ_rst5 āgrahāyaṇī māse.

  - आरम्भदेशः मापने (starting point of place) (rsp5)
    Example: तिरुपतितः_rsp5 चन्द्रगिरिः क्रोशे ।
    tirupatitaḥ_rsp5 candragiriḥ krośe.

  - लक्षणम् (point of direction) (rd)
    Example: वृक्षं प्रति_rd विद्योतते विद्युत् ।
    vṛkṣaṁ prati_rd vidyotate vidyut.
    पक्षी भवनस्य उपरि_rd डयते ।
    pakṣī bhavanasya upari_rd dayate.
    रामः ग्रामं प्रति_rd गतः ।
    rāmaḥ grāmaṁ prati_rd गतः ।

  - तादर्थ्य (purpose) (rt)
    Example: बालकाय_rta पुस्तकं क्रीणाति ।
    bālakāya_rt pustakaṃ krīṇāti.

  - हेतुः hetuḥ (rh)
    Example: कुम्भकारः दण्डेन_rh घटं करोति ।
    kumbhakāraḥ daṇḍena_rh ghaṭaṃ karoti.

- सह सम्बन्धः (associative) (ras)
(पुत्रेण सह)_ras पिता गच्छति।
(putreṇa saha)_ras pitā gacchati.

- विना (non-associative) (rnas)
(धर्मेण विना)_rnas जीवनं नास्ति।
(dharmeṇa vinā)_rnas jīvanaṃ nāsti.

- विभक्तः (comparison between two) (rv5)
माथुराः पाटलीपुत्रकेभ्यः_rv5 आढ्यतराः।
māthurāḥ pāṭalīputrakebhyaḥ_rv5 āḍhyatarāḥ

- निर्धारणम् (Isolating one from a group – in the superlative degree context) (rn7 / rn6).
गवां_rn7 कृष्णा बहुक्षीरा।
gavāṁ_rn7 kṛṣṇā bahukṣīrā
गोषु_rn6 कृष्णा बहुक्षीरा।
goṣu_rn6 kṛṣṇā bahukṣīrā

Appendix

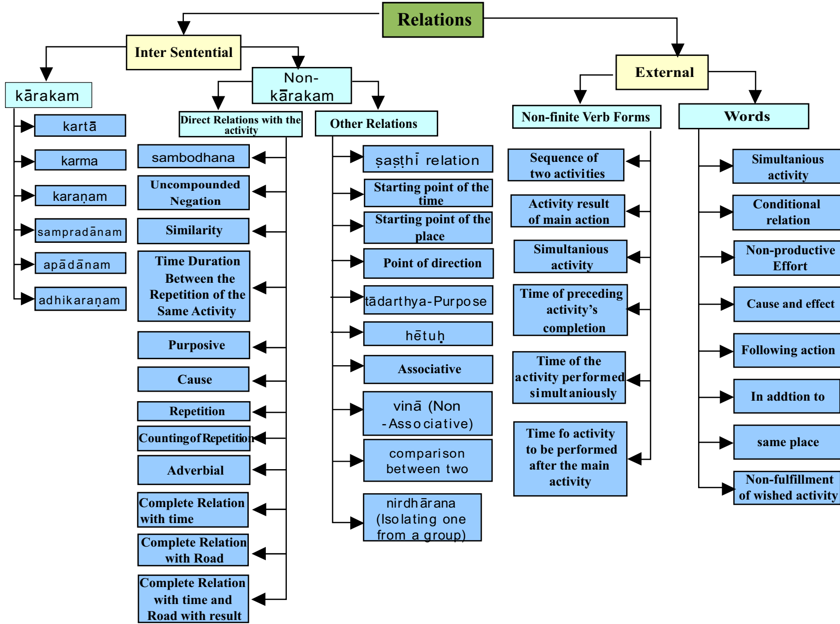


Fig. 1. Relations in anvaya-prakriyā

Modelling the Grammatical Circle of the Pāṇinian System of Sanskrit Grammar

Anand Mishra

Department of Classical Indology
Ruprecht Karls University, Heidelberg, Germany
amishra@ix.urz.uni-heidelberg.de

Abstract. In the present article we briefly sketch an extended version of our previously developed model for computer representation of the Pāṇinian system of Sanskrit grammar. We attempt to implement an antecedent analytical phase using heuristical methods and improve the subsequent phase of reconstitution using the rules of *Aṣṭādhyāyī* by incorporating strategies for automatic application of grammatical rules.

Keywords: Sanskrit, Grammar, Pāṇini, Aṣṭādhyāyī, Computational Linguistics, Modelling.

1 Introduction

In the following we propose a computer model for representing the grammatical process of the Pāṇinian system of Sanskrit grammar. In Sect. 2 we describe the circular nature of this grammatical process. It consists of first an analysis of a given linguistic expression into constituent elements and then its reconstitution using a given set of rules. This process, thus starts with a provisional statement and ends in a *saṃskṛta* or perfected expression.

A brief summary of our model for implementing the latter step of reconstitution of linguistic expressions using the rules of *Aṣṭādhyāyī* is presented in Sect. 3. The implementation of the preceding analytical step is now being attempted. The main approach here is to improvise heuristics to guess the constituent elements of a given expression. This is outlined in Sect. 4. The subsequent reconstitutive step as specified by *Aṣṭādhyāyī* is now proposed to include strategies for automatic application of rules in Sect. 5. The main program modules implementing all this are mentioned in Sect. 6.

2 Circular Nature of the Grammatical Process

The *Aṣṭādhyāyī* of Pāṇini is a device to produce linguistic expressions using its constituent elements. It prescribes a set of fundamental components which constitute the language, characterizes them using a number of attributes and specifies rules to form the final linguistic expressions. This process of constructing

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 40–55, 2009.

linguistic expressions presupposes a process of analysis, as a result of which, Pāṇini[1] had at his disposal fundamental components like *bhū*, *ti*P[2] etc.

The analytical process, beginning with the *pada-pāṭha* of Vedic *mantra*s is not recorded in terms of rules of analysis. Thus, MADHAVA DESHAPANDE [3] notes that Pāṇini's grammar "does not provide us with analytical tools to go from texts to their interpretation, or from sentences to morphemes and phonemes. It presupposes the results of an analytical phase of scholarship, but is itself not representative of that analytical phase".

The *Aṣṭādhyāyī* records processes only of the second phase of a *grammatical circle* that begins with a provisional expression which a speaker formulates to express her/his intention (*vivakṣā*) and culminates in formation of a *saṃskṛta* expression. The first phase, that of analysis of a sentence in one or more *pada*s (*vākyavibhajyānvākhyāna*) and further a *pada* in its constituting components i.e. *prakṛti* and *pratyaya* etc. (*padavibhajyānvākhyāna*), precedes the process of synthesis.[3]

Thus, the grammatical process can be stated as consisting of the following steps:

1. Collection of *pada*s P_j from a (provisional) sentence S_i.

$$S_i = \{P_j\}_1^n \tag{1}$$

   Decomposition of a *pada* P_j into *prakṛti* (root) R and *pratyaya*s (affixes) $A_{1\cdots k}$.

$$P_j = \{R, A_{1\cdots k}\} \tag{2}$$

2. Combination of the constituent components into *pada*s and sentence.

$$\{R, A_{1\cdots k}\} \longrightarrow P'_j \tag{3}$$

$$\{P'_j\}_1^n \longrightarrow S'_i \tag{4}$$

The above steps comprise a circular process of first decomposing a (provisional) sentence into *imaginary*[4] components and then reassembling these components to form a *saṃskṛta* expression.

$$S_i \longrightarrow \{P_j\}_1^n \longrightarrow \{R, A_{1\cdots k}\} \longrightarrow \{P'_j\}_1^n \longrightarrow S'_i \tag{5}$$

Thus, modelling the grammatical process involves modelling this circular process of decomposition and then recombination, through which a provisional sentence is transformed into a *saṃskṛta* sentence.[5]

---

[1] Here, a reference to Pāṇini includes his predecessor, contemporary and successor grammarians as well.

[2] The *it* - markers are represented in SMALL CAPS.

[3] See BHATTACHARYA [1] Pp. 228.

[4] See BHATTACHARYA [1] Pp. 229.

[5] See HOUBEN [7] Pp. 48.

3 Representing the Grammatical Process of *Aṣṭādhyāyī*

This section briefly summarizes the basic data structure for modelling the grammatical process of *Aṣṭādhyāyī*.[6]

3.1 Fundamental Components

The building blocks of the language are collected and assigned a unique key in our database. For example, the phoneme /*a*/ has the key `a_0`, the *kṛt* suffix /*a*/ has the key `a_3` and the *taddhita* suffix /*a*/ is represented by the key `a_4`.

Definition 1. *The collection of unique keys corresponding to the basic constituents of the language, we define as the set $\mathcal{F}$ of fundamental components.*

Remark 1. This set is further sub-divided in two disjoint sets consisting of the set of keys corresponding to the phonemes ($\mathcal{P}$) and the set containing the keys of the rest of the constituting elements ($\mathcal{M}$).

$$\mathcal{P} = \{\texttt{a_0}, \texttt{i_0}, \texttt{u_0}, \ldots\} \tag{6}$$

$$\mathcal{M} = \{\texttt{bhU_a}, \texttt{tip_0}, \texttt{laT_0}, \ldots\} \tag{7}$$

3.2 Attributes

The fundamental units of the language are given an identity by assigning a number of attributes to them. This includes the various technical terms introduced in the grammar as also the *it* -markers and *pratyāhāra*s.

Definition 2. *The collection of unique keys corresponding to the terms, which characterize a fundamental component, we define as the set $\mathcal{A}$ of attributes.*

Remark 2. Corresponding to the sets $\mathcal{P}$ and $\mathcal{M}$ we can decompose the set $\mathcal{A}$ into two disjoint sets $\mathcal{A}_\pi$ and $\mathcal{A}_\mu$, $\mathcal{A}_\pi$ being the set of unique keys of the attributes to the elements of $\mathcal{P}$ and $\mathcal{A}_\mu$ to elements of $\mathcal{M}$.

$$\mathcal{A} = \mathcal{A}_\pi \cup \mathcal{A}_\mu \tag{8}$$

$$\mathcal{A}_\pi = \{\texttt{hrasva_0}, \texttt{udAtta_0}, \texttt{it_0}, \ldots\} \tag{9}$$

$$\mathcal{A}_\mu = \{\texttt{dhAtu_0}, \texttt{pratyaya_0}, \texttt{zit_9}, \ldots\} \tag{10}$$

Remark 3. Any two of the four sets $\mathcal{P}, \mathcal{M}, \mathcal{A}_\pi, \mathcal{A}_\mu$ are mutually disjoint.

Given the mutually disjoint sets $\mathcal{P}$, $\mathcal{M}$ and $\mathcal{A}$, we represent a linguistic expression at any stage of its derivation through a *language component*, which is an ordered collection of *sound sets*. We first define a sound set and then a language component in terms of these sound sets.

[6] For a detailed description, see Mishra [9].

3.3 Sound Set ψ

Definition 3. *A sound set ψ is a collection of elements from sets $\mathcal{P}, \mathcal{M}$ and $\mathcal{A}$ having exactly one element from the set $\mathcal{P}$.*

$$\psi = \{\pi_p, \mu_i, \alpha_j | \pi_p \in \mathcal{P}, \mu_i \in \mathcal{M}, \alpha_j \in \mathcal{A}, i, j \geq 0\} \tag{11}$$

3.4 Language Component λ

Definition 4. *A language component λ is an ordered collection of at least one or more sound sets.*

$$\lambda = [\psi_0, \psi_1, \psi_2, \dots \psi_n] \text{ such that } \|\lambda\| > 0 \tag{12}$$

A *language component* λ has as many *sound sets* ψ_i's as there are phonemes in that component. A *sound set* ψ contains a number of fundamental components and attributes. Those attributes which are common to a number of sound sets in a language component, become the attributes of that chain of sound sets. This chain could be a single phoneme, or a morpheme or more than one morphemes and even more than one words.

The process of formation is represented through a *process strip* σ, which is an ordered collection of a pair having its first entry as a rule number and second one to be a language component, which is achieved after application of this rule.

3.5 Process Strip σ

Definition 5. *A process strip σ is an ordered collection of pairs, where the first element of the pair is the number of a particular grammatical rule (e.g. $rule_p$) and the second element is a language component λ.*

$$\sigma = [(rule_p, \lambda_p), (rule_q, \lambda_q), \dots] \tag{13}$$

There are two basic operations, *attribute addition* and *augmentation* which are applied to a language component. All the operations in *Aṣṭādhyāyī* e.g. substitution, reduplication, accentuation etc. are implemented using a combination of these two basic operations.

3.6 Attribute Addition

Let $\alpha \subset \mathcal{A} \cup \mathcal{M}$ and ψ be a sound set. Then attribute addition is defined as

$$h_{a\psi}(\psi, \alpha) = \psi \cup \alpha \tag{14}$$

Remark 4. This operation can be applied to a number of sound sets given by indices $[i, i+1, \dots, j]$ in a given language component λ

$$h_{a\lambda}(\lambda, \alpha, [i, \dots, j]) = [\psi_1, \dots, \psi_i \cup \alpha, \dots, \psi_j \cup \alpha, \dots, \psi_n] \tag{15}$$

3.7 Augmentation

Let

$$\lambda = [\psi_1, \ldots, \psi_i, \psi_{i+1}, \ldots, \psi_n]$$
$$\lambda_k = [\psi_{1k}, \psi_{2k}, \psi_{3k}, \ldots, \psi_{mk}]$$

and i be an integer index such that $i \leq \|\lambda\|$, then augmentation of λ by λ_k at index i is defined as

$$h_g(\lambda, \lambda_k, i) = [\psi_1, \ldots, \psi_i, \psi_{1k}, \psi_{2k}, \psi_{3k}, \ldots, \psi_{mk}, \psi_{i+1}, \ldots, \psi_n] \tag{16}$$

3.8 Substitution

We define substitution in terms of the above two operations.

Let $[i, i+1, i+2, \ldots, j]$ be the indices of sound sets to be replaced in the language component $\lambda = [\psi_1, \ldots, \psi_i, \psi_{i+1}, \ldots, \psi_n]$.

Let $\lambda_k = [\psi_{1k}, \psi_{2k}, \psi_{3k}, \ldots, \psi_{mk}]$ be the replacement, then the substitution is defined as

$$h_s(\lambda, \lambda_k, [i, \ldots, j]) = h_g(h_{a\lambda}(\lambda, \{\delta\}, [i, \ldots, j]), \lambda_k, j) \tag{17}$$

where $\delta \in \mathcal{A}$ is the *attribute* which says that this sound set is no more active and has been replaced by some other sound set.

A rule of grammar is represented through a function f_q, which takes a process strip σ_p and adds a new pair $(rule_q, \lambda_q)$ to it where $rule_q$ is the number of the present rule, and λ_q is the new modified language component after application of one or more of the two basic operations defined above on the input language component λ_p.

$$f_q(\sigma_p) = \sigma_q \text{ where} \tag{18}$$
$$\sigma_p = [\ldots, (rule_p, \lambda_p)] \tag{19}$$
$$\sigma_q = [\ldots, (rule_p, \lambda_p), (rule_q, \lambda_q)] \tag{20}$$
$$\lambda_q = h_a, h_g(\lambda_p, \ldots) \tag{21}$$

A typical formative process begins with a *seed* element (usually a verbal root or nominal stem), and a chain of rules provided manually through a template is applied. At the end, a *saṃskṛta* expression is formed.

The system has been tested for different formative processes of *Aṣṭādhyāyī* and can be accessed online (`http://sanskrit.sai.uni-heidelberg.de`).

4 Heuristically Analyzing the Sanskrit Expressions

Aṣṭādhyāyī provides us with a collection $\mathcal{F}$ of fundamental elements which is a finite set, having limited entries. Using this set and another finite collection of rules, a substantially bigger set of linguistic expressions can be formed. To

specify this process, a number of meta-linguistic entities (collected in set $\mathcal{A}$ of attributes) as well as conventions are used.

This process presupposes another process of looking for these fundamental components in the expressions of the language. For example, by some process of analysis, it is ascertained that with *bhavati* the elements like *bhū* or Ś*a*P or *ti*P are associated.[7] As mentioned earlier, there are no recorded rules for this step. The question, which fundamental elements are associated with a particular expression, can however be approached heuristically.[8] The problem can be stated as follows:

Problem 1. Given a string S, search for the possible break-up tuples such that each tuple contains only fundamental elements which could be later used as seeds for reconstitution.

The above task is performed by an `Analyzer (A)` which aims at *guessing* the possible fundamental components constituting a given sentence S. It does not aim to ascertain the perfect break up in terms of fundamental constituents, but only some of the possible components, which could function as seeds for the subsequent step of reconstitution.

Example 1. We give an example first for a sub-problem, where our string S is a *pada*. Given $S = jayati$, `A`(S) should fetch us at least a tuple consisting of at least the verbal root *ji* and possibly the *ti*Ṅ suffix *ti*P as well.

$$\mathtt{A}(jayati) = [(ji, ti\text{P}, \ldots), (e_1, \ldots), (e_2, \ldots), \ldots] \quad \text{where } e_i \in \mathcal{F} \tag{22}$$

In fact, it gives a list of possible decomposition tuples.

4.1 Some Observations on the Process of Analysis

Before we describe our approach for developing heuristics towards analysing an expression and give an example, we first mention a few observations as to the nature of this problem.

On the surface, it seems to be searching for a needle in a hay stack, but a closer look allows for such an adventure! For this purpose, certain features of the grammatical corpus and processes of *Aṣṭādhyāyī* can be made use of.

1. The set of fundamental elements $\mathcal{F}$ is finite. That means, we do not have to search infinite elements.
2. The order of fundamental elements in a tuple is also not random. Thus, (*upasarga*, *dhātu*, *vikaraṇa*, *pratyaya*) is one such (partial) order.

[7] The examples here are at *pada* level and not at *vākya* level, although the unit of analysis (as well as synthesis) is a sentence. This is because of simplicity and also it does not amount to any loss of generality.

[8] I am also working on the possibilities to incorporate some statistical methods, but it is too early to report about it.

3. Simultaneous presence of certain fundamental elements within a tuple is also restricted. For example, while analysing a *pada*, both a *ti*Ṅ suffix as well as a *s*UP suffix can not occur simultaneously.
4. Certain attributes of the fundamental elements, like *avasāna*, *s*UP, *ti*Ṅ indicate *pada* boundaries. This is helpful to identify more than one *pada*s within a character string.
5. A dictionary of possible surface forms (as keys) and corresponding original elements (as values) provides a connection between phoneme chains on the surface level to the fundamental elements, which may have contributed to it. Here, a special sub-set is of those elements, like *dhātu*s or *ti*Ṅ suffixes which are more abundantly present.
6. Consonants are less prone to phonetic changes.
7. The replacement rules (*ādeśa-sūtra*s) can be reversed, and these reversed rules can be used to gain the replaced element. Thus, for example, the replacement rule **thā aḥ se** (3.4.080)[9] replaces the whole of *thās* of a *ṭit lakāra* with *se*. So in case, *se* is identified, the reversal of this rule will be used to check the possibility of *thās* here.
8. Reverse engineering of certain standard *vidhi*s, e.g. reduplication or *ṣatva vidhi* etc. brings us closer to the original element.

Finally it should be mentioned that it is for the teleological purpose of providing *seeds* for the subsequent step of reconstitution, with which this phase is concerned and not to provide a correct and complete decomposition of a sentence or a word. In fact an imprecise break up of an incorrect *pada* can only possibly lead to the *saṃskṛta* form.

4.2 The General Process of Analysis

Given a character string, the general process of analysis involves in guessing its possible break ups in terms of elements of the set $\mathcal{F}$ of fundamental constituents. It consists of the following steps:

1. Take a possible break up of character string.
2. Try to find the corresponding elements associated with these sub-strings using the dictionary which maps surface forms to fundamental elements.
3. Try to find the possible replaced elements using reverse replacement rules in a given tuple of fundamental elements.
4. Check for Pāṇinian consistency of this tuple.
5. If consistent, then add to the list of break up tuples.
6. Repeat the previous steps for a new initial break up.

We illustrate the above process through a couple of examples.

Example 2. Consider *pavete* to be the input string. We first try to guess the possible *ti*Ṅ suffix. For that, we look up in the the dictionary which maps surface forms to fundamental elements for *ti*Ṅ suffixes. This dictionary looks like

$$\{ti : [ti\text{P}, \ldots], ta\d{h} : [tas, \ldots], \ldots, te : [\bar{a}t\bar{a}m, \ldots], \ldots\}$$

[9] Numbers in brackets refer to the rule number in *Aṣṭādhyāyī*.

We take only those break up strings which can possibly be associated to some *ti*Ṅ element. Thus, the possible break up is restricted through the keys of the dictionary of surface forms to fundamental elements. In this case, we break the string as *pave te* and associate *ātām* to *te*. We represent this as follows

$$[(pave)(te : \bar{a}t\bar{a}m)]$$

Next we look at the leading part (*pave*) of the provisional break up, which has the possibility that it may contain the verb. Here we look first in the list of verbs beginning with *p*. These are [*pac*I, *paṭ*A, ..., *puṣ*A, *pū*Ṅ, *pū*Ñ, ...]. We now use the rules for reverse replacement. This is guided by the standard replacement series in verbs. One such series is $\bar{u} \rightarrow o \rightarrow av$ replacements. The character string *av* motivates the reverse replacement, and applying this, we come from (*pave*) to (*pūe*). We associate now the verbs *pū*Ṅ as well as *pū*Ñ to the sub-string *pū*. We thus have,

$$[(p\bar{u} : p\bar{u}\dot{\text{N}}, p\bar{u}\tilde{\text{N}})(e)(te : \bar{a}t\bar{a}m)]$$

We now collect the decomposition tuples. These are,

$$[(p\bar{u}\dot{\text{N}}, \bar{a}t\bar{a}m), (p\bar{u}\tilde{\text{N}}, \bar{a}t\bar{a}m), \ldots]$$

Now the Pāṇinian consistency of the tuples are checked. In this case the tuples are (*dhātu*, *pratyaya*) tuples. So the order of elements within a tuple is correct. Moreover, within a tuple, there is no simultaneous presence of mutually exclusive pairs of fundamental elements. For example, no *s*UP and *ti*Ṅ suffixes are present simultaneously. Thus, these two tuples are added to the list of other possible break up tuples of fundamental elements.

$$L = [\ldots, (p\bar{u}\dot{\text{N}}, \bar{a}t\bar{a}m), (p\bar{u}\tilde{\text{N}}, \bar{a}t\bar{a}m), \ldots]$$

The process is repeated for other possible character break ups (as long as there is such a possibility). The tuples are ranked according to the richness of information they contain for the subsequent process of reconstitution. Thus, those tuples, having a *dhātu* or *prātipadika* and a *ti*Ṅ or *s*UP suffix are ranked higher than those having only a *dhātu* etc.

Example 3. Consider the case where the input string (for a *pada*) has four or more consonants. We look for the possibility whether it is the case of reduplication, specially because of the suffix *sa*N.

1. Look whether there is consonant *s* or *ṣ* in the input string
2. Use the heuristics for deciding the *ti*Ṅ endings for a *pada* (see previous example) and check if *s* or *ṣ* appear before these.
3. Get the part before *s* or *ṣ* and check it with heuristics for reduplication.

Now the heuristics of reduplication is implemented taking care of the process of reduplication in *Aṣṭādhyāyī*. Thus, let us consider the case, where input has three consonants.

1. The probability of a root with two consonants is high.
2. Get the list of roots having the consonants as the last two consonants of input. (Here, also the roots which undergo changes due to *ṇatva* or *ṣatva vidhi* etc.)
3. Check if the first consonant of input could be a reduplicated consonant e.g. pairs like *j*-*g* or *c*-*k* etc.
4. If the last consonant is *r* or *l* then consider the possibility of *ṛ* or *ḷ*.

Consider now the input string: *titikṣate*. The process of analysis is briefly sketched below.

1. Consider a possible string break up: *titik ṣa te*
2. Check heuristics for *ti*Ṅ and assume that it returns (*ta*, *ātām*. . .)
3. Split at *s* or *ṣ*: *titik ṣate*
4. Send the first half before *s* or *ṣ* to check for reduplication.
5. We have here three consonants: *t t k*
6. Search for the roots with consonants *t k* (as also the possible variants *t j*, *t g* etc.). It returns roots like *tik*A, *tig*A, *tij*A, *tuj*A, *tuj*I etc.
7. Check the first consonant of the input, if it is according to the reduplication rules.
8. Now reduce the choice further using other heuristics (e.g. looking at the vowels between the two consonants of the root).
9. Thus, the output of this heuristics is: [(*tik*A, *sa*N, *ta*), (*tig*A, *sa*N, *ta*), (*tij*A, *sa*N, *ta*), . . .]

5 Forming *Saṃskṛta* Expressions Using the Rules of *Aṣṭādhyāyī*

Given a break up tuple, the process of forming the *saṃskṛta* expression(s) is regulated by the rules of *Aṣṭādhyāyī*. We developed a model for constituting linguistic expressions beginning with seed elements and through manual prescription of rule order using templates (see Sect. 3). We now propose to include strategies for automatic application of rules.

5.1 An Extended Model for Forming *Saṃskṛta* Expressions

This part of the grammatical process is being implemented in the `Synthesizer` module, which takes as input a tuple of fundamental elements. This tuple is gained by the preceding step of analysis. All the constituent elements for formation of a particular expression are not provided in this input tuple, but only the *seed* elements, e.g. verbal root or nominal stem and if possible *ti*Ṅ or *s*UP suffixes etc. This initial tuple contains partial information about the *vivakṣā* or intention of the speaker.

Further, the input tuple is *consistent*. Consistency means that it contains only those elements which can occur simultaneously. Moreover, the ordering of

these elements is according to the Pāṇinian principles. For example, the element corresponding to *dhātu* must precede the *pratyaya* or suffix element.

In the `Synthesizer`, the elements of this input tuple are taken as *seeds* and a number of appropriate rules are applied with the goal of reproducing the *saṃskṛta* form of the original expression. For this purpose, the data structure and corresponding operations are described in Sect. 3. The entire process is simulated using the process strip. All the information which is required to assess the conditions for application of a particular rule is stored in this process strip, which stores not only the current stage of formation but also the previous stages.

The question, which rule must be applied next, was resolved thus far by prescribing a template based approach in which the order of rules to be applied was stated manually. We now propose strategies for automatic application of rules. For this, we introduce stable and transitional λ - states.

5.2 Stable and Transitional $\boldsymbol{\lambda}$ - States

At any stage of formation, the fundamental components together with their attributes are represented in a language component λ. Given such a language component, we first try to bring it in a *stable* λ - state by applying certain rules which we call stabilizing rules.

The purpose of this step is to prepare the current λ - state for assessing the 'cause of application' or *nimitta* of those transitional rules which bring about a transition of λ - state. We first specify what we mean by stabilizing and transitional rules.

Stabilizing Rules. There are certain rules in *Aṣṭādhyāyī* (specially most of the definition rules), which need to be applied to a λ - state in order to add more grammatical information which is necessary for a progressive flow of the process of formation of linguistic expressions.

For example, if a new element is introduced in the previous step, which contains the phoneme /*ā*/, then the application of rule **vṛddhirādaic** (1.1.001) adds the information that it also has the attribute *vṛddhi*, which may be required for subsequent application of other rules. Similar rules which bring about some kind of attribute addition are what we call stabilizing rules.[10]

We collect these stabilizing rules seperately and define this set as follows:

Definition 6. *The set of stabilizing rules* $\mathcal{R}_\int$ *is the set of those characterizing rules in* Aṣṭādhyāyī, *which fulfill the condition that the application of any rule belonging to this set on a language component is not depended upon the results of application of any other rule of this set.*

For example, the characterizing rules **vṛddhirādaic** (1.1.001) and **adeṅguṇaḥ** (1.1.002) belong to the set of stabilizing rules $\mathcal{R}_\int$.

[10] In fact many fundamental elements have certain attributes which are *static*, i.e. they are always associated with that particular fundamental element. For example, with the element /*a*/ the attribute *a*C (specifying that it is a vowel) is always attached.

Having defined the set of stabilizing rules, we can now speak of a *stabilizing process* ($\longrightarrow$) which brings a language component λ to a stable language component λ' by applying the stabilizing rules from the rule set $\mathcal{R}_{\int}$.

$$\lambda_i \longrightarrow \lambda'_i \tag{23}$$

Transitional Rules. Those rules which do not belong to the set of stabilizing rules, we call transitional rules. The effect of the application of a particular rule belonging to this set has a consequence for the application of some other rule belonging to this same set. Barring those characterizing rules grouped under $\mathcal{R}_{\int}$, all the other rules, we put in this group.

The process of transition of λ - states caused by application of transitional rules can now be considered as a *transitional process* ($\Longrightarrow$).

$$\lambda_i \Longrightarrow \lambda_{i+1} \tag{24}$$

The General Formative Process. The general Pāṇinian process of formation of linguistic expressions can now be presented as an incremental increase of process strip σ through a transitional phase and then stabilization of the strip through a stabilizing phase, whereby the two phases always alternate.

$$[\ldots,([rule_{p_i}],\lambda_p \longrightarrow \lambda'_p)] \Longrightarrow [\ldots,([rule_{p_i}],\lambda'_p),([rule_{q_i}],\lambda_q \longrightarrow \lambda'_q)] \tag{25}$$

5.3 Executing the Stabilizing Process

The stabilizing phase ($\lambda_p \longrightarrow \lambda'_p$) is executed every time by applying the rules from the set $\mathcal{R}_{\int}$. This helps in characterizing the current situation of the language component.

Example 4. For example, if a morpheme like Ś*a*P is added in the previous transitional phase, the following stabilizing phase adds the attributes like *hrasva*, *guṇa*, *śit*, *pit*, *sārvadhātuka* etc. to the sound set corresponding to the phoneme /*a*/ of Ś*a*P.

5.4 Executing the Transitional Process

Given a stable λ - state within a process strip, the main challenge here is to decide as to which rule should next be applied to proceed through the transitional step. A correct and definite answer to this problem is the key for automatic application of rules in the process of formation of linguistic expressions according to *Aṣṭādhyāyī*. The problem can be divided into two sub-steps.

1. What are the possible rules which could be applied?
2. Given more than one possibilities, how to choose the correct one?

Assessing a $\boldsymbol{\lambda}$ - State. A sub-module `Assessor` assesses a given λ - state and gives a tuple of list of rules, which could be applied for a transitional process. This tuple is then sent to `ConflictResoluter`, another sub-module, which evalutes

the question of conflicting rules and fetches those options which may lead to correct results. If there are more than one possibilities, then all are pursued in a parallel manner.

The `Assessor` Module. There are two guiding principles here to decide which rules can possibly now be applied:

1. Assessment of the intention (*vivakṣā*) of the speaker.
2. Assessment of the thus far evolution of formative process, i.e. assessment of the input process strip.

Assesing the Intention of the Speaker. One way to assess the intention of the speaker is to provide a user interface at this level. But for now, we depend on our heuristic analysis of the original provisional input by the speaker.

Example 5. If *bhavati* is what the speaker inputs and if our analysis provides us with one such tuple like (*bhū*, *ti*P) then we can guess some of the information as to what she/he wants to convey. Thus, at some stage when the question arises which *ti*Ṅ suffix is to be attached, then the entry in the input tuple can give us the information.

Example 6. Given the initial tuple (*tij*A, *sa*N, *ta*), and the situation that we have just the *dhātu* in the stable $\lambda - state$ (λ'), the question as to which way to proceed could be answered by looking at the presence of *sa*N.

Assesing the Stable $\boldsymbol{\lambda}$ − *state*. The assessment of the stable λ - states in a process strip is based upon the observations as to what are the elements which are currently present and what could be the next introduction (*āgama*) or substitution (*ādeśa*). Here, certain guidelines for the general flow of the formative process are first taken into consideration.

For example, in the situation where only a *dhātu* is present and the input tuple has a *ti*Ṅ suffix, the general flow would be to apply rules for introduction of *lakāra*. If there is already a *lakāra* and no *ti*Ṅ substitute, then rules for such a substitution are applied.

The observations regarding the general order of introduction of fundamental elements are stored in terms of defining a partial order of these elements. This partial ordering aims to provide the answer to the question as to which element should next be introduced.

Certain morphemes give rise to a scope of applying certain rules once they are introduced. This observation is collected in the form of a special dictionary, where the possible rules, which could subsequently be applied, are listed. For example *l*IṬ, *sa*N etc. trigger reduplication, and so the rules which bring about reduplication are listed with these morphemes.

Conflict Resolution. The successive filtering of possible candidates should normally provide a definite answer to the question of rule application. But in some cases, there are conflicting claims. Here we follow the solutions according

to the principles laid down by JOSHI-ROODBERGEN [5]. One such principle for a 'one-way conflict' is that in case of rules not belonging to *asiddhakāṇḍa*, "that rule is to be applied first, which destroys the *nimitta* of the other rule, or which changes the phonetic form to which the other rule was to become applicable".[11]

The `ConflictResoluter` Module. The conflict resolution algorithms are implemented in the module `ConflictResoluter` which gets as input, the process strip together with a tuple of list of conflicting rules. It checks the applicability of these rules and returns the one which should be applied. We briefly show its functioning by way of one example.

Example 7. Consider the reconstitutive process of *dudyūṣati* and let the process strip correspond to the stage[12]

$$\sigma_p = di\bar{u} + sa\text{N} + \acute{S}a\text{P} + ti\text{P}$$

At this stage, the `Assessor` proposes two possibilities ([6.1.009], [6.1.077]). The first one is the rule **san yaṅ oḥ** (6.1.009) which calls for reduplication and the second one is the rule **ikaḥ yaṇ aci** (6.1.077) which prescribes substitution of *ya*Ṇ respectively in place of *i*K. The `ConflictResoluter` now checks as follows:

1. Take the process strip σ_p and apply **san yaṅ oḥ** (6.1.009). This means applying the process of reduplication. This gives the extended process strip with the new λ - state as
$$\sigma_q = di + di\bar{u} + sa\text{N} + \acute{S}a\text{P} + ti\text{P}$$
2. Now this new λ - state is stabilized and then checked using `Assessor` if the conflicting rule (6.1.077) is still applicable. This is the case here. So, the application of this rule neither changes the *nimitta* of the conflicting rule nor does it bring about any change in the phonetic form of the elements to which the other rule is to be applied.
3. The other rule **ikaḥ yaṇ aci** (6.1.077) is now applied to the process strip σ_p. This gives the result
$$\sigma_q = dy\bar{u} + sa\text{N} + \acute{S}a\text{P} + ti\text{P}$$
4. The resulting new λ - state is assessed and it shows that the phonetic form of the element to which the other rule is to be applied has been changed.
5. This results in selection of the rule **ikaḥ yaṇ aci** (6.1.077) for application at this stage, because both the rules do not belong to the range of rules of *asiddhakāṇḍa*.

Similarly, other principles of conflict resolution are implemented. We enunciate below the general architecture of the computer implementation of the modelling process. At the moment, the modules are in the programming phase and testing of a wider range of examples are needed before the system could be put to use.

[11] JOSHI-ROODBERGEN [6] Pp. X.

[12] The process strip actually is a complex data structure which is expressed in terms of language components, which in turn is a list of sound sets (see Sect. 3), but for the sake of simplicity, we write it here in this form.

6 Computer Implementation of the Model

The entire process is divided into four main modules besides a specially designed `Database` of the fundamental components $\mathcal{F}$ and attributes $\mathcal{A}$ of Pāṇinian Grammar. These four main modules are:

1. `Input`
2. `Analyzer`
3. `Synthesizer`
4. `Output`

Each of them contain a number of other sub-modules. We sketch briefly the main ones below.

6.1 Database

Besides having a repository of the fundamental components $\mathcal{F}$ and attributes $\mathcal{A}$ in *Aṣṭādhyāyī*, there are a few special dictionaries and lists of elements and attributes serving specific purposes. It includes

1. A dictionary which maps surface forms to fundamental elements, which is used by the `Analyzer` module. It looks like:

   {*ti* : [*ti*P, . . .], *taḥ* : [*tas*, . . .], . . . , *te* : [*ātām*, . . .], . . .}

2. A set of pairs whose elements exclude each other within an analysis tuple. E.g. {(*s*UP, *ti*Ṅ), (*l*AṬ, *l*AṄ), . . . }
3. A list of acceptable partial orders within an analysis tuple. E.g. [(*upasarga*, *dhātu*, *vikaraṇa*, *pratyaya*), (*dhātu*, *sa*N, *vikaraṇa*), . . .]
4. A dictionary, used by `Assessor`, of fundamental elements as keys and list of rules which are possibly applied when this element is introduced as values.
5. A number of special subsets of the set of fundamental elements or attributes for the sake of identification of respective elements. For example, the set of phoneme attributes, or the set of morpheme attributes etc.

6.2 Input

This module, as the name suggests, takes care of the user interface for entering a provisional sentence S. After an initial processing, e.g. checking the non-occurence of phonemes not belonging to the language, the input is passed to the `Analyzer`.

6.3 Analyzer (See Sec. 4)

Given a sentence S, the `Analyzer` aims at *guessing* the possible fundamental components constituting it. The `Analyzer` functions heuristically and suggests

first a number of possible break ups. These possibilities are then ranked based upon certain constraints and provide *seeds* for `Synthesizer`.

Thus, given a string S, the `Analyzer (A)` produces a ranked list D of possible decompositions, represented as tuples containing the fundamental components constituting S.

$$\mathtt{A}(S) = D = [(e_1, e_2, \ldots), (e_3, \ldots), (e_4, \ldots), \ldots] \quad \text{where } e_i \in \mathcal{F} \tag{26}$$

6.4 Synthesizer (See Sec. 5)

Given an analysis tuple $t = (e_1, e_2, e_3, \ldots)$, the `Synthesizer (Z)` now applies a series of rules from *Aṣṭādhyāyī*, which are collected in a rule set $\mathcal{R}$, and produces the final expression S'.

$$\mathtt{Z}((e_1, e_2, e_3, \ldots)) = S' \quad \text{where } e_i \in \mathcal{F} \tag{27}$$

For the purpose of assessing a given stage during the process of formation, it uses the `Assessor` module, which outputs a tuple of lists of rules which could be applied at that stage. In case of a conflict of rules, the `ConflictResoluter` module tries to resolve the clash. Otherwise all the possibilities are checked in a parallel manner.

6.5 Output

This module outputs the reconstituted sentence S' as well as the original provisional sentence S. It also provides a step-by-step process of constitution of the final expression beginning with its elemental parts and grammatical information gathered during the course of formation.

References

1. Bhattacharya, R.S.: Pāṇinīya Vyākaraṇa kā Anuśīlana. Indological Book House, Varanasi (1966)
2. von Böhtlingk, O.: Pāṇini's Grammatik, Olms, Hildesheim. Primary source text for our database (1887)
3. Deshpande, M.M.: Semantics of Kārakas in Pāṇini: An Exploration of Philosophical and Linguistical Issues. In: Matilal, B.K., Bilimoria, P. (eds.) Sanskrit and Related Studies: Contemporary Researches and Reflections, pp. 33–57. Sri Satguru Publications, Delhi (1990)
4. Dīkṣita, P.: Aṣṭādhyāyī sahajabodha, vols. 1-4. Pratibha Prakashan, Delhi (2006-2007)
5. Joshi, S.D., Roodbergen, J.A.F.: On siddha, asiddha and sthānivat Annals of the Bhandarkar Oriental Research Institute, vol. LXVIII, Poona, pp. 541–549 (1987)
6. Joshi, S.D., Roodbergen, J.A.F.: The Aṣṭādhyāyī of Pāṇini. With Translation and Explanatory Notes, vol. II. Sahitya Akademi, New Delhi (1993)

7. Houben, J.E.M.: ‘Meaning statements’ in Pāṇini’s grammar: on the purpose and context of the Aṣṭādhyāyī. Studien zur Indologie und Iranistik 22, 23–54 (1999) [2001]
8. Katre, S.M.: Aṣṭādhyāyī of Pāṇini. Motilal Banarsidass, Delhi (1989)
9. Mishra, A.: Simulating the Pāṇinian System of Sanskrit Grammar. In: Proceedings of the First International Sanskrit Computational Linguistics Symposium, Rocquencourt, pp. 89–95 (2007)
10. Śāstrī, C.: Vyākaraṇacandrodaya, vols. 1-5. Motilal Banarsidass, Delhi (1971)
11. Vasu, S.C., Vasu, V.D.: The Siddhānta-Kaumudī of Bhaṭṭojī Dīkṣita. vols. 1-3. Panini Office, Bhuvanesvara Asrama, Allahabad, India. Primary source text for prakriyā (1905)

Computational Structure of the *Aṣṭādhyāyī* and Conflict Resolution Techniques

Sridhar Subbanna[1] and Shrinivasa Varakhedi[2]

[1] Rashtriya Sanskrit Vidyapeetha, Tirupati, India
sridharsy@gmail.com
[2] Samskrita Academy, Osmania University, Hyderabad, India
shrivara@gmail.com

Abstract. *Pāṇini*'s *Aṣṭādhyāyī* consists of sūtras that capture fundamentals of Sanskrit language and define its structure in terms of phonology, morphology and syntax. The *Aṣṭādhyāyī* can be thought of as an automaton to generate words and sentences. In object oriented programming terms, prescribing sūtras are objects having its transformation rule as its method. The meta rules or paribhāṣā sūtras and paribhāṣā vārtikās define the flow of the program. During application of sūtras, conflicts may arise among two or more competing sūtras. In this paper, computational structure of the *Aṣṭādhyāyī*, sūtra objects observing the environment, tree representation of sūtras and mathematical representation of conflict resolution techniques are presented.

Keywords: Pāṇini, Aṣṭādhyāyī, Sanskrit, Vyākaraṇa, Sūtra, Computer Modelling, Conflict Resolution, Object Oriented Programming, Mathematical Representation.

1 Introduction

The *Aṣṭādhyāyī*[1] (śabdānuśāsanam) deals with the generation of words and sentences of Sanskrit Language and also provides a base for the analysis of the same in general. Its algebraic nature and comprehensiveness illustrate that its structure can be described as a machine generating words and sentences of Sanskrit.

The *Aṣṭādhyāyī*[9] consists of around 4000 sūtras that describe the fundamentals of Sanskrit language in terms of phonology, morphology and syntax. The structure consists of definitions, rules, and meta-rules that are context-sensitive and operate in sequence or recursively[6].

Generally these rules are classified into three groups:

1. Rules of definition and meta-rules (saṃjña and paribhāshā)
2. Rules of affixation of roots (dhātu and prātipadika) and
3. Rules of transformation for stems and the suffixes.

The computer programs have exactly the same features of context-sensitive rules, recursion and sequential rule application[7]. Prescribing sūtras are context-sensitive rules, paribhāṣās define the flow and hence the *aṣṭādhyāyī* can be thought of as an automaton that generates Sanskrit words and sentences.

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 56–65, 2009.

2 Computational Structure of the *Aṣṭādhyāyī*

The *Aṣṭādhyāyī* consists of sūtras that are organized in a systematic and mathematical manner. The application of sūtras follows a systematic procedure in deriving the final form of words from roots and affixes. In object oriented programming[8], objects have state and behaviour as two characteristics. State is the data part and behaviour is the method or function. Here, sūtra as an object will be observing the state of environment and applies as its transformation rule if the condition satisfies. Thus, object oriented programming suits best to model the aṣṭādhyāyī.

2.1 Classification and Representation of Sūtras

Traditionally the sūtras in the *Aṣṭādhyāyī* are classified into 6 groups.[1]

1. *saṃjñā sūtras*: assign saṃjña-s or labels.
2. *paribhāṣā sūtras*: meta-rules for interpretion and application of sūtras.
3. *vidhi sūtras*: prescribing rules for ādeśa(substitution/deletion) and āgama(insertion).
4. *niyama sūtras*: conditioning rules that confines vidhi sūtra with some additional conditions.
5. *atideśa sūtras*: extension rules to extend the existing rules in another scenario.
6. *adhikāra sūtras*: governing rules adopted for division of contents and gives meaning to the successive sūtras.

For our convenience these sūtras can be grouped as follows.

1. Meta rules or helping rules (2 & 6 above)
2. Prescribing rules (1,3,4 & 5 above)

paribhāṣā and *adhikāra* sūtras are meta rules. *Paribhāṣā* sūtras are utilized to intrepret the sūtras and *adhikāra* sūtras are meant to define the boundary of a particular topic domain. Rest are prescribing rules that define transformation functions. The meta rules help in interpretation[4] and application of rules. This will be discussed in detail in the conflict resolution section.

The prescribing rules can be grouped under each nested topic (*ekavākya*). Each nested group can be represented as a tree of sūtra objects. *utsarga* or general sūtra will be root node and *apavāda* (exception) sūtras will be the child nodes. The sūtras that are having different conditions under the same topic will be the sister nodes. During the application of sūtras a single tree traversal algorithm can be used to determine the sūtra that is to be applied. It has to be seen whether this process of tree building can be automated.

The following examples will explain the nature of representation.

Example 1: The Figure 1 shows representation of mutually exclusive *guru* and *laghu saṃjñas.*

[1] *saṃjñā ca paribhāṣā ca vidhir niyama eva ca*
atideśodhikāraśca ṣaḍvidham sūtralakṣaṇam.

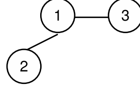


Fig. 1. Tree representation for *guru* and *laghu saṃjña* rules

1. *hrashvam laghu 1.4.10*
2. *saṃyoge guru 1.4.11*
3. *dhīrghaṃ ca 1.4.12*

Example 2: At various different places the *it saṃjña* is used. The Figure 2 shows its definition.

1. *upadeśe ajanunāsika it 1.3.2*
2. *halantyam 1.3.3*
3. *na vibhaktau tusmāḥ 1.3.4*
4. *ādiḥ ñiṭudavaḥ 1.3.5*
5. *ṣaḥ pratyayasya 1.3.6*
6. *cutū 1.3.7*
7. *laśakvataddhite 1.3.8*

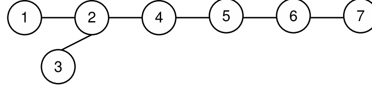


Fig. 2. Tree representation for *it saṃjña* rules

Example 3: The Figure 3 shows the representation of the sandhi sūtras. The sūtra 2, 3, 4 & 6 are sister nodes as their conditions for applying or domain are different. The sūtra 12 is apavāda to both sūtra 2 and sūtra 6. There may be many such cases in whole of the *aṣṭādhyāyī*.

1. *saṃhitāyāṃ 6.1.72*
2. *iko yaṇaci 6.1.77*
3. *ecoyavāyāvaḥ 6.1.78*
4. *vānto yi pratyaye 6.1.79*
5. *dhātostannimittasaiva 6.1.80*
6. *ādgunaḥ 6.1.87*
7. *vṛddhireci 6.1.88*
8. *etyedhatyūṭhsu 6.1.89*
9. *eṅi pararūpaṃ 6.1.94*
10. *omāngośca 6.1.95*
11. *ato gune 6.1.97*
12. *akaḥ savarṇe dīrghaḥ 6.1.101*
13. *prathamayoḥ pūrvasavarṇaḥ 6.1.102*

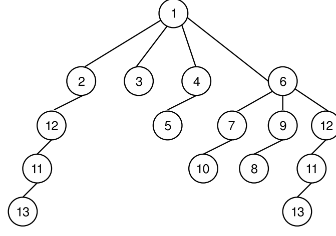


Fig. 3. Tree representation for *ac sandhi* rules

2.2 Computational Structure

The root nodes will be observing the environment (subject). All the nodes that find the condition send a request for their application. After getting requests, conflict resolver adopts the resolution techiques and selects only one node among them for application. Then the tree with that as root node will be traversed to find the exact sūtra for application. The sūtra object contains all the information about the sūtra. The sūtra will be applied to update the environment. The Figure 4 represents the overall structure of the aṣṭādhyāyī.

The output states and other sūtras are siddha (visible) to all the sūtras of the *aṣṭādhyāyī* except where it is explicitly mentioned as asiddha (invisible). There are three places where the asiddhatva is explicitly mentioned.

1. *Asiddhavadatrābhāt 6.4.22*
2. *Ṣatvatukorasiddhaḥ 6.1.86*
3. *Pūrvatrāsiddham 8.2.1*

In general, whenever conditions for a sūtra are satisfied, that sūtra will be applied. Only one sūtra will be applied at a time. In other words no two sūtras can be applied simultaneously. However, the sūtras of asidhavat prakaraṇa that are all having the same condition, can be thought of as applied simultaneously. The sūtras in the last three pādas, that is, tripādī need to be applied in the sequential order. The Figure 4 explains this model.

In Figure 4, the siddha block contains the sūtras of sapādasaptādhyāyī minus the asiddhavat prakaraṇa sūtras (6.4.22 to 6.4.175). The asiddhavat block contains the sūtras of asiddhavat prakaraṇa. The asiddha block contains the tripādi sūtras. The ovals A and B represent the output states and the C represents the output of the system (final state). The small circles represent sūtra group that takes the input state and changes the state. There will be a state transition function defined for each sūtra object. While in a particular state, if the conditions for root sūtra are satisfied, then complete tree with this as a root node is traversed in order to find out which sūtra within this tree is to be applied, accordingly it transforms the state by invoking the function defined for that sūtra. In the siddha block, all the states are visible and can be input state to any sūtra in the same block.

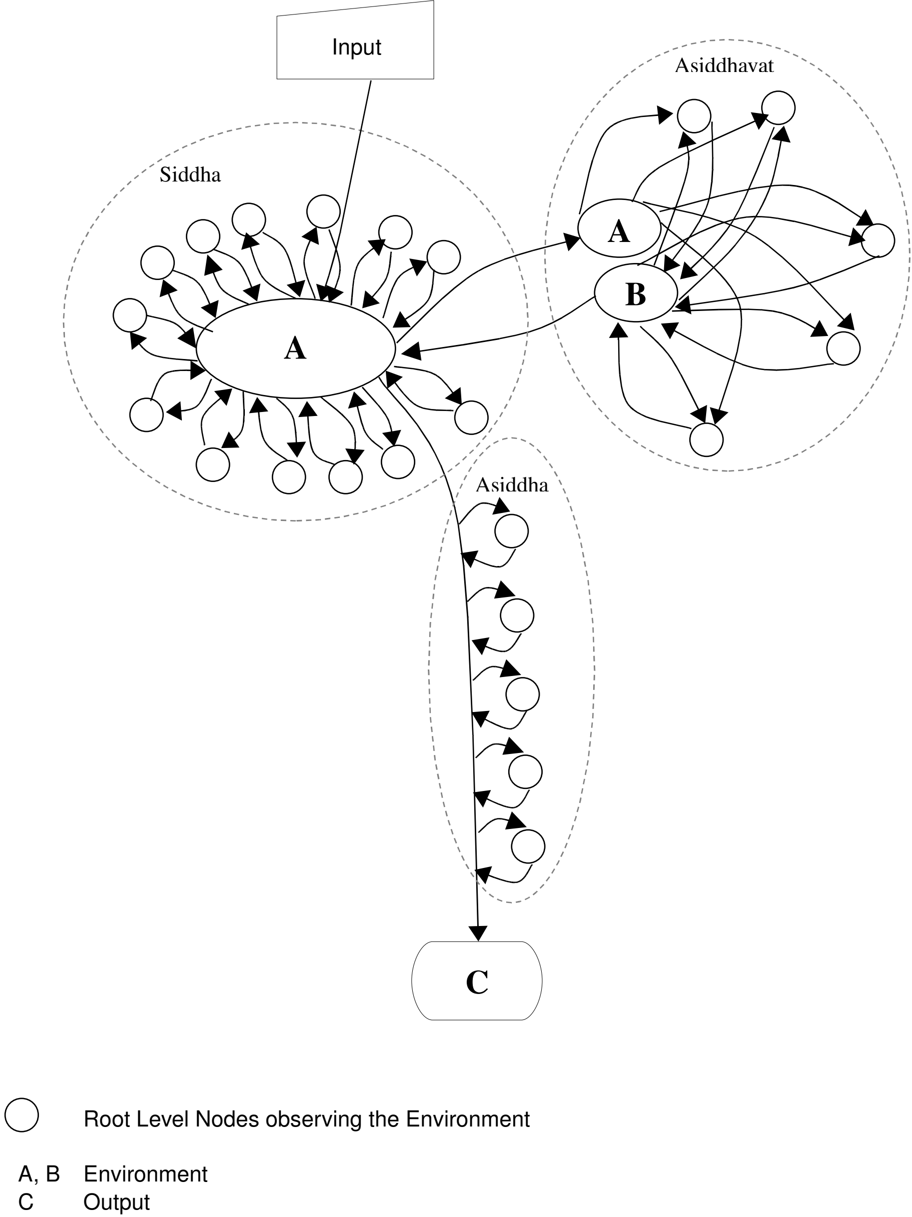


Fig. 4. Computational Structure of the *Aṣṭādhyāyī*

While in the siddha blok, if conditions of any of the sūtras in the *asiddhavat* block are met, initially, state B is same as state A. The sūtras here in the asiddhavat block take both A and B as input, and transforms only B. So, all other sūtras check for their condition in state A and operates on B. When

there are no more conditions for the sūtras in this block, the state A is made same as state B.

When there are no more conditions for sūtras in the siddha block or in the *asiddhavat* block then the state is sequentially updated by the sūtras in asiddha block that are applicable in that particular state and the final output state will be C.

There is a need to develop techniques that make the explanation of the computational structure of the *aṣṭādhyāyī* easier. The techniques of representing and manipulating knowledge should be developed and create computing algorithms that have computational abilities.

Example 1. When the state A is *vana + ṭā*(case 3,num 1)], the sūtra *ṭā ṅasiṅasām inātsyāḥ* (7.1.12) in the siddha block finds its condition. After its invocation A is changed to *vana + ina*. Then the sūtra *ād gunaḥ* (6.1.87) in the siddha block finds its condition and gets invoked. A now becomes *vanena*. No other sūtra finds the condition, hence A is passed to C. Now C has the final form *vanena*.

Example 2. When the state in A is *śās + hi* sūtras in the Asiddhavat block find the condition. Initially, state B will be same as state A. Now the sūtra *śā hau* (6.4.35) is invoked and then B is changed to *śā + hi*. The sūtra *hujhalbhyo herdhiḥ* (6.4.101) also finds the condition in A and is invoked. B is now changed to *śā + dhi*. No more sūtras finds the condition, so A is made same as B. None of the sūtras either in siddha block or the asiddha block find the condition in A. Hence A is passed to C without any tranformation giving the final word [śādhi].

Example 3. When the state A is *dvau + atra*. The sūtra *ecoyavāyāvaḥ* (6.1.78) finds the condition. This sutra is now applied, A is changed to *dvāv + atra*. No sūtra in the siddha block or the asiddhavat block finds condition in A. The sūtra *lopaḥ śākalyasya* (8.3.19) in the asiddha block finds the condition. After its invocation it is changed to *dvā + atra*. This state is not visible to the sutras either in the siddha block or in the asiddhavat block. Even though there is condition for *akaḥ savarṇe dīrghaḥ* (6.1.101) in the siddha block, this state is not visible to this sūtra. Hence this sūtra cannot be applied. The sūtras that are following the sūtra *lopaḥ śakalyasya*(8.3.19) have visibility of this state but do not find condition. Hence none of the sūtras are applied, yielding the final form as *dvā atra*.

The asiddhatva is the base for the above shown computational structure of the aṣṭādhyāyī. This whole structure developed on the basis of Siddha-asiddha principle resolves many conflicts in the whole application.

3 Techniques for Conflict Resolution

In general, it can be thought of, as all the sūtras will be observing the changes in a given state (prakriyā) and wherever they find their condition (nimitta), they

would come forward to apply their function - kārya on that state, as a result the state would get modified. However there are possibilities of conflict that arise between many sūtras in this process, as many of them may find condition in a particular state and all of them would try to apply. At any point of time only one modification is possible in the given state. Sometimes all the sūtras may be applied in a particular order or one is applied and others are rejected. Due to this, complexity of the program will not only increase, but also results into a paradoxical situation.

There are different paribhāṣā sūtras, paribhāṣā vārtikās to resolve these conflicts. Considering only the *aṣṭādhyāyī* sūtras it may not be possible to resolve conflicts under all circumstances, hence vārtikas also should be taken into account in conflict resolution. We are trying to represent conflict resolution techniques mathematically, that can be directly adopted in a computer program.

3.1 Conflict Resolution through Sūtras

There are only few sūtras that directly prescribe the flow or resolve conflict.

3.1.1 *Vipratiṣedhe paraṃ kāryaṃ*

1.4.2 This sūtra says when there is conflict then para (the later one in a sequential order) sūtra is to be applied. According to Patanjali's interpretation *para* means *iśṭa*[2] and not the later one in the sūtra order. There is another controversy among traditional and modern Grammarians in interpretation of this sūtra. The traditional view is that this sūtra is globally applicable across the aṣṭādhyāyī. The modern thinking is that this sūtra is locally applicable to the *ekasaṃjñā* domain which runs through 2.2.38. This needs to be looked into greater detail to see the cases on which they have taken their stands.

3.1.2 *Siddha* and *Asiddha*

All the sūtras are treated as *siddha* (visible) to each other unless specified explicitly as *asiddha* (invisible) in the sūtras.[2]

There are two view points on *asiddha* concept namely *Śāstrāsiddha* and *Kāryāsiddha*. If a sūtra is *Śāstrāsiddha*, if sūtra itself is invisible to another sūtra, and it is *Kāryāsiddha* if the sūtra is visible but its *kārya* (result) is *asiddha* to the other. These two alternative ideas need to be examined.

Example *śivacchāyā*[10]

1. *śivachāyā*
2. *śivatchāyā 6.1.73*
3. *śivadchāyā 8.2.39*
4. *śivajchāyā 8.4.40*
5. *śivacchāyā 8.4.55*

[2] *pūrvatra asiddham 8.2.1, asiddhavadatrābhāt 6.4.22, ṣatvatukorasiddhaḥ 6.1.86.*

The environment is *śivachāyā*. The sūtra *che ca 6.1.73* is applied and is changed to state 2. Now the sūtras *jhalāṃjaśonte* 8.2.39 and *stoḥścunāścuḥ* 8.4.40 find the condition in this context. Since 8.4.40 is asiddha to 8.2.39, 8.2.39 is applied first and state is updated to state 3. Again, 8.4.40 and *khari ca* 8.4.55 has nimmita for application, similar to earlier one, here also since 8.4.55 is asiddha to 8.4.40, 8.4.40 is applied first and state is update to state 4. Now, 8.4.55 gets a chance for its application and environment is moved to state 5. The sūtra *coḥ kuḥ* 8.2.30 cannot see the environment and does not come forward. This way application of rule 8.2.30 is prevented after the final form.

3.2 Conflict Resolution through *vartikas*

3.2.1 *Para-nitya-antaraṅga-apavādānām uttarottaraṃ balīyaḥ*

[3] This paribhāṣā gives us criterion for conflict resolution. The priority is *apavāda, antaraṅga, nitya* and *para*. We explain below how we model these priorities.

1. *utsarga - apavāda*
   utsarga and *apavāda* (General and Exception) sūtras are static; this information is embedded in the tree structure itself. During the application of sūtras the tree is traversed in such a way to determine the apavāda sūtra. When two sūtras have *utsarga* and *apavāda* relation then *apavāda sūtra* is selected and applied, *utsarga sūtra* is rejected.

2. *antaraṅga - bahiraṅga*
   The definition *alpāpekṣaṃ antaraṅgaṃ* and *bahvapekṣaṃ bahiraṅgaṃ* can be used to determine the *antaraṅgatva* and *bahiraṅgatva* of the any two sūtras. When the sūtras have *antaraṅga* and *bahiraṅga* relation then *antaraṅga sūtra* is selected and applied, *bahiraṅga sūtra* is rejected. The *antaraṅga* and *bahiraṅga* are relative to the context and can be mathematically determined. The definition could be formalized as follows.
   Let $f(X, \phi)$ return the number of conditions that are required for sūtra X to apply in the given state ϕ.
   if $f(X, \phi)$ is less than $f(Y, \phi)$
   then X is *Antaraṅga* and Y is *Bahiraṅga*
   else Y is *Antaraṅga* and X is *Bahiraṅga*
   endif

   Example, Let X = *sarvādīni sarvanāmāni* 1.1.27 and Y=*prathamacharama tayalpakatipayanemāśca* 1.1.33. When the state is *ubhaya jas*. Whether *ubaya* gets *sarvanāmasaṃjñā* by X or optionally by Y is the question.
   $f(X, \phi) = 1$ as there is only one condition for X to apply. The condition is ubhaya's existance in *sarvādi gana*.
   $f(Y, \phi) = 2$ as there are two conditions for Y to apply. One condition is *ubhaya* is a *tayappratyayānta* and second condition is *jas pratyaya* in front of it.

Since, $f(X,\phi) < f(Y,\phi)$, X is *antaraṇga* and Y is *bahiraṇga.* Since *antaraṇga* is preferred, here *ubhaya* will get the *sarvanāmasaṃjñā* by X, but not optional *sarvanāmasaṃjñā* by Y.

3. *Nitya - Anitya*
The definition of *nitya* and *anitya* is given as *kṛtākṛtaprasaṅgi nityam tadviparītamanityam.* The *nitya* and *anitya* are relative to the context and can be mathematically defined.
Let there be sūtras X and Y that have the condition for its application in a particular state. If X is applicable irrespective of the application of Y then X is said to be *nitya.* On application of Y if X loses its condition for application then it is said to be *anitya.* It can be defined mathematically as follows.
Let $f(X,\phi)$ returns $\phi\prime$,transformed state after application of sūtra X in the ϕ state and returns zero if sūtra X is not applicable in the ϕ state.

if $f(X,f(Y,\phi))$ is not zero
X is *nitya* and Y is *anitya*
else

if $f(Y,f(X,\phi))$ is not zero
X is *anitya* and Y is *nitya*
else X and Y do not have the *nitya anitya* relation.

Comsider the case when the state is *tud ti* then the sūtras *tudādibhyaḥ śaḥ*(3.1.77) and *pugantalaghūpadasya ca* (7.3.86) both find their condition. Let X = *tudādibyaḥ śaḥ(3.1.77)* and Y = *pugantalaghūpadasya ca*(7.3.86). Then $f(X,\phi) =$*tud sha ti* and $f(Y,f(X,\phi)) = 0$ because Y is not applicable in the state *tud sha ti.* So Y is anitya. Consider, $f(Y,\phi) =$ *tod ti* and $f(X,f(Y,\phi)) =$*tod sha ti* i.e, is not equal to zero. Hence X is nitya.

4. *Para - Apara*
Para and Apara (Posterior and Prior): The sūtra that is positioned before in the *aṣṭādhyāyī* order is *apara* and the one later is *para.* Between the *para* and *apara* sūtras, *para sūtra* is selected and applied, apara sūtra is rejected. The para-apara relation can be determined based on the *sūtra saṅkhyā.*
For example, *bahuvacane Jhalyet* 7.3.103 is *para* to *supi ca* 7.3.102 as 7.3.103 is later than 7.3.102. Let $f(X)$ returns the sūtra number in the aṭādhyāyī.
if $f(X) > f(Y)$
then X is *para* and Y is *apara*
else Y is *para* and X is *apara*

3.2.2 *Varṇādāṅgaṃbalīyo bhavati*

There are two rules one acting on aṅga[3] and other on phoneme. In this case, the sūtra on aṅga should be applied first.

[3] *yasmāt pratyayavidhiḥ tadādi pratyaye aṅgam - 1.4.13.*

3.2.3 *Lakṣye lakṣaṇaṃ sakṛdeva pravartate*

This vartika prevents the recursion. Only once any sūtra should be applied in a particular environment[5]. Here sūtra means group of sūtras under the same topic (ekavākya). In the tradition, analogy of *takrakauṇḍiṇya*[4] is given to expain this concept.

4 Conclusion

The next objective is to implement this as a computer program and see whether we can optimize the sūtras or evaluate the necessity of the vārtikas. Such an implimentation would not only confirm the automata nature of Paninian System but also exhibit the complexities of the system and feasibility of resolutions to them by employing techniques shown by Pāṇinian Tradition. Our current effort could be a first step towards achieving that goal.

Acknowledgment. Authors thank Lalit Kumar Tripati and Amba P Kulkarni for useful discussions at various stages of the work.

References

[1] Giri, S.P., Satyanarayanashastri: Paṇinīyaḥ Aṣṭādhyāyī Krishnadas Academy, Varanasi (1984)
[2] Josi, B.S.B.: The Vyākaraṇamahābhāṣya of Patañjali. Chaukhamba Sanskrit Pratishtan, Delhi (1872)
[3] Kielhorn, F.: Paribhāṣenduśekara of Nāgojibhaṭṭa. Parimal Publications, Delhi
[4] Mishra, S.: Kāśikā. Chaukamba Samskrita Samsthan, Varanasi (1979)
[5] Abhyankar, K.V.: Paribhāshāsamgraha. Bhandarkar Research Instt., Puna (1967)
[6] Kiparsky, P.: On the Architecture of Panini's Grammar. Conference on the Architecture of Grammar, Hyderabad (2002)
[7] Bhate, S., Kak, S.: Panini's Grammar and Computer Science, Annals of the Bhandarkar Oriental Research Institute, 79–94 (1993)
[8] Roy, P.V., Haridi, S.: Concepts, Techniques and Models of Computer Programming. MIT Press, Cambridge (2004)
[9] Vasu, S.C.: The Aṣṭādhyāyī of Pāṇini. Motilal Banarasidas Publishers, New Delhi (2003)
[10] Vasu, S.C.: Siddhānta kaumudi. Motilal Banarasidas Publishers, New Delhi (2002)

[4] *Brāhmaṇebhyo dadhi dīyataṃ, takraṃ kauṇḍiṇyāya.* Here there are two rules. *Brāhmaṇebyo dadhi dīyatāṃ* is first one, *takraṃ kauṇḍiṇyāya* is the second one. Once *takraṃ* is given to *kauṇḍiṇya*, again *dadhi* will not be given according to first rule.

Levels in Pāṇini's *Aṣṭādhyāyī*

Peter M. Scharf

Department of Classics, Brown University, PO Box 1856, Providence, RI 02912
scharf@brown.edu

Abstract. In 1969 Kiparsky and Staal proposed that Pāṇini's *Aṣṭādhyāyī* contained a four-level hierarchy of rules. While modifying the interrelation of the levels, Kiparsky (2002) still maintains the four-level hierarchy. R. Rocher (1964: 51) and Cardona (1976: 215-224) argued against such a hierarchy, the former maintaining that Pāṇini operated just with a two-level hierarchy of meaning and speech. Cardona was willing to accept the propriety of speaking of one intermediate level on the grounds that the assignment of kāraka terms involved both semantic and cooccurrence conditions. The present paper clarifies the issue, argues that the assignment of abstract *l*-affixes to the same level as kāraka classification by Kiparsky is problematic, that most rules considered to be purely phonetic (sandhi rules) in fact include morphological conditions and concludes that although there are intermediate stages in derivation, Pāṇini considers there to be just two levels. The semantic and syntactic levels are properly coalesced in a syntacticosemantic level and the abstract morphological and the morphophonemic level are properly coalesced in a single morphophonemic level.

Keywords: levels, generative grammar, Panini, Astadhyayi, phonology, morphology, syntax, semantics, morphophonemic, syntacticosemantic, computational implementation.

1 Kiparsky's Architecture

One of the most prominent contemporary linguistic models used to interpret Pāṇinian grammar is the idea that grammar consists of modules in a generative hierarchy, or levels. Clearly influenced by Chomskian generative grammar, Kiparsky and Staal (1969) proposed that Pāṇinian grammar contains rules in a hierarchy of four levels of representation: semantics, deep structure, surface structure, and phonology. More recently Kiparsky (2002) restates this scheme referring to the four levels as follows: (1) semantic, (2) morphosyntactic, (3) abstract morphological, and (4) phonological (see Fig. 1). Three classes of rules map prior levels onto subsequent levels: (1) rules that assign kārakas and abstract tense, (2) morphological spellout rules, and (3) rules of allomorphy and phonology. Rules incorporate conditions at both the levels from which and to which they map, as well as at prior levels in a unidirectional derivation beginning with semantics and ending with phonology.

As an example of how derivation is understood to work in the four-level hierarchy, one may take the derivation of the sentence *Devadatta odanaṁ pacati* (Fig. 2). At the semantic level, the speaker intends to express that Devadatta, called here John Doe,

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 66–77, 2009.

undertakes the action of cooking in present time for the purpose of making boiled rice. Pāṇinian semantics classifies John Doe as the independent agent in the action, and boiled rice as that which is desired to be obtained. Four rules apply to map the semantic level onto the morphosyntactic level. 1.4.49 and 1.4.54 assign kārakas, 3.4.69 lets an *l*-affix occur to denote an agent (*kartṛ*), and 3.2.123 assigns abstract tense by introducing the *l*-affix *laṭ* on the condition that present time is to be denoted.

1. Semantic information
   ↓ Assignment of kārakas (th-roles) and of abstract tense
2. Morphosyntactic representation
   ↓ Morphological spellout rules
3. Abstract morphological representation
   ↓ Allomorphy and phonology
4. Phonological output form

Fig. 1. Levels according to Kiparsky 2002: 3

1. John Doe$_{\text{[svatantra]}}$ rice$_{\text{[īpsitatama]}}$ cooks$_{\text{[vartamāna]}}$.
   John Doe$_{\text{[independent]}}$ rice$_{\text{[desideratum]}}$ cooks$_{\text{[present]}}$.

   1.4.49 *kartur īpsitatamaṁ karma*
   ↓ 1.4.54 *svatantraḥ kartā*
   3.4.69 *laḥ karmaṇi ca bhāve cākarmakebhyaḥ*
   3.2.123 *vartamāne laṭ*

2. Devadatta$_{\text{[kartṛ]}}$ odana$_{\text{[karman]}}$ ḍupacaṣ+laṭ.
   Devadatta$_{\text{[agent]}}$ odana$_{\text{[direct object]}}$ pac+laṭ.

   3.4.78 *tiptasjhi...iḍvahimahiṅ*
   1.3.78 *śeṣāt kartari parasmaipadam*
   1.4.108 *śeṣe prathamaḥ*
   1.4.22 *dvyekayor dvivacanaikavacane*
   ↓ 3.1.68 *kartari śap*
   4.1.2 *svaujasamauṭ...ṅyossup*
   2.3.2 *karmaṇi dvitīyā*
   2.3.46 *prātipadikārthaliṅgaparimāṇavacanamātre prathamā*

3. Devadatta+su odana+am ḍupacaṣ+śap+tip.
   Devadatta+$_{\text{[nom]}}$ odana+$_{\text{[acc]}}$ pac+$_{\text{[3sa pre]}}$.

   1.3.9 *tasya lopaḥ*
   6.1.107 *ami pūrvaḥ*
   ↓ 8.3.17 *bhobhagoaghoapūrvasya yo 'śi*
   8.3.19 *lopaḥ śākalyasya*
   8.3.23 *mo 'nusvāraḥ*

4. Devadatta odanaṁ pacati.
   Devadatta cooks rice.

Fig. 2. Example of Four-level Derivation

Several "spellout" rules then apply to map the morphosyntactic level onto the abstract morphological level. 3.4.78 provides that a basic verbal termination replaces the *l* of the affix ***laṭ*** that occurs after the verbal root *pac*. Restrictive rules 1.3.78, 1.4.108 and 1.4.22, read in conjunction with 3.4.78, select the third person singular active (3sa) affix ***tip*** on condition that a single agent that is neither the speaker nor the addressee is to be denoted. Before the affix ***tip*** (termed *sārvadhātuka* by 3.4.113 *tiṅśit sārvadhātukam*), 3.1.68 provides the default verbal stem-forming affix ***śap*** to cosignify the agent. Then 4.1.2 provides nominal terminations. Restrictive rules 2.3.2, 2.3.46, and 1.4.22, read in conjunction with 4.1.2 select the appropriate nominal termination. 2.3.2 selects a second triplet nominal termination (*dvitīyā*) after the stem *odana* on condition that the kāraka *karman*, which has not yet been denoted (*anabhihite* 2.3.1), is to be denoted. 2.3.46 selects a first triplet nominal termination (*prathamā*) after the stem *devadatta* on condition that just the stem meaning, gender, and number are to be denoted. (The kāraka *kartr̥* has already been denoted by the verbal termination thus preventing 2.3.18 *kartr̥karaṇayos tr̥tīyā* from applying.) 1.4.22 selects the singular terminations *am* (2s) and ***su*** (1s), respectively in each triplet.[1]

Finally, several rules of allomorphy (of which there are none in the present example) and phonology apply to map the abstract morphological level onto the phonological level.[2]

The example of the derivation of the sentence *Brāhmaṇāya phalāny adāt*, shown in Fig. 3, provides greater detail. At the semantic level, the speaker intends to express that someone, indicated by an X, undertakes the action of giving in past time for the purpose of transferring his ownership of fruit to a Brahmin. Pāṇinian semantics classifies X as the independent one in the action, and the fruit as that which is desired to be obtained. Five rules apply to map the semantic level onto the morphosyntactic level. 1.4.32, 1.4.49 and 1.4.54 assign kārakas, 3.4.69 lets an *l*-affix occur to denote an agent (*kartr̥*), and 3.2.110 assigns abstract tense by introducing the *l*-affix ***luṅ*** on the condition that past time is to be denoted.

Several "spellout" rules then apply to map the morphosyntactic level onto the abstract morphological level. 3.4.78 provides that a basic verbal termination replaces the *l* of the affix ***luṅ*** that occurs after the verbal root *dā*. Restrictive rules 1.3.78, 1.4.108 and 1.4.22, read in conjunction with 3.4.78, select the third person singular active (3sa) affix ***tip*** on condition that a single agent that is neither the speaker nor the addressee is to be denoted. Before the affix ***tip*** (termed *sārvadhātuka* by 3.4.113 *tiṅśit sārvadhātukam*), 3.1.43 provides the default abstract verbal stem-forming affix ***cli*** that occurs with verbal terminations that replace ***luṅ***. Then 4.1.2 introduces nominal terminations. Restrictive rules 2.3.2, 2.3.13, and 1.4.22, read in conjunction with 4.1.2 select the appropriate nominal terminations. 2.3.2 selects a second triplet nominal termination (*dvitīyā*) after the stem *phala* on condition that the kāraka *karman*, which has not yet been denoted (*anabhihite* 2.3.1), is to be denoted. 2.3.13 selects a fourth triplet nominal termination (*cathurthī*) after the stem *brāhmaṇa* on condition

[1] Rules 1.4.99-108 that designate verbal and nominal terminations in the lists 3.4.78 and 4.1.2 by terms that allow selection according to person, number, and voice are not shown.

[2] The rule that deletes markers, 1.3.9, is shown here though its application is simultaneous with the introduction of affixes.

that the kāraka *saṁpradāna* is to be denoted. 1.4.21 selects the plural second-triplet termination *śas* (2p) after the stem *phala*, and 1.4.22 selects the singular fourth-triplet termination *ṅe* (4s) after the stem *brāhmaṇa*.

1. $\text{X}_{[\text{svatantra.eka}]}$ $\text{Brahmin}_{[\text{karmaṇā yam abhipraiti.eka}]}$ $\text{fruit}_{[\text{īpsitatama.bahu}]}$ $\text{gave}_{[\text{bhūta.śeṣa.eka}]}$.
   $\text{X}_{[\text{independent.one}]}$ $\text{Brahmin}_{[\text{whom one intends with the direct object.one}]}$ $\text{fruit}_{[\text{desideratum.many}]}$ $\text{gave}_{[\text{past.3rdperson.one}]}$.

   1.4.32 *karmaṇā yam abhipraiti sa saṁpradānam*
   1.4.49 *kartur īpsitatamaṁ karma*
   ↓ 1.4.54 *svatantraḥ kartā*
   3.4.69 *laḥ karmaṇi ca bhāve cākarmakebhyaḥ*
   3.2.110 *luṅ* (*bhūte* 84)

2. $\text{Brāhmaṇa}_{[\text{saṁpradāna.eka}]}$ $\text{phala}_{[\text{karman.bahu}]}$ $\text{ḍudāñ+luṅ}_{[\text{kartṛ.bhūta.śeṣa.eka}]}$.
   $\text{Brāhmaṇa}_{[\text{indirect object.one}]}$ $\text{phala}_{[\text{direct object.many}]}$ $\text{dā+luṅ}_{[\text{3rdperson.one}]}$.

   3.4.78 *tiptasjhi...iḍvahimahiṅ*
   1.3.78 *śeṣāt kartari parasmaipadam*
   1.4.108 *śeṣe prathamaḥ*
   1.4.21 *bahuṣu bahuvacanam*
   ↓ 1.4.22 *dvyekayor dvivacanaikavacane*
   3.1.43 *cli luṅi*
   4.1.2 *svaujasamauṭ...ṅyossup*
   2.3.2 *karmaṇi dvitīyā*
   2.3.13 *caturthī saṁpradāne*

3. $\text{Brāhmaṇa+ṅe}_{[\text{caturthī.ekavacana}]}$ $\text{phala+śas}_{[\text{dvitīyā.bahuvacana}]}$ $\text{ḍudāñ+cli+tip}_{[\text{luṅ.prathama.ekavacana}]}$.
   $\text{Brāhmaṇa+}_{[\text{dative.sg}]}$ $\text{phala+}_{[\text{accusative.pl}]}$ $\text{dā+}_{[\text{3sa aor}]}$.

   1.3.9 *tasya lopaḥ*
   3.4.100 *itaś ca* (*ṅitaḥ* 99, *lasya* 77, *lopaḥ* 97)
   3.1.44 *cleḥ sic*
   1.4.99 *laḥ parasmaipadam*
   2.4.77 *gātisthāghupābhūbhyaḥ sicaḥ parasmaipedeṣu* (*luk* 58)
   7.1.13 *ṅer yaḥ*
   ↓ 7.1.20 *jaśśasoḥ śiḥ* (*napuṁsakāt* 19)
   1.1.42 *śi sarvanāmasthānam*
   7.1.72 *napuṁsakasya jhalacaḥ* (*num* 58)
   7.3.102 *supi ca* (*ato dīrgo yañi* 101)
   6.4.8 *sarvanāmasthāne cāsaṁbuddhau* (*nopadhāyāḥ* 7, *dīrghaḥ* 6.3.111)
   6.4.71 *luṅlaṅlṛṅkṣv aḍ udāttaḥ*
   6.1.77 *iko yaṇ aci*

4. Brāhmaṇāya phalāny adāt.
   He gave fruit to the Brahmin.

Fig. 3. Fuller Example of Four-level Derivation

Finally, several rules of allomorphy and phonology apply to map the abstract morphological level onto the phonological level. Three rules modify the basic terminations provided after the verbal and nominal stems: 3.4.100 deletes the *i* in the basic verbal termination *ti* that replaces *luṅ*, 7.1.13 replaces the basic singular fourth triplet nominal termination *ṅe* with *ya*, and 7.1.20 replaces the basic plural second triplet nominal termination *śas* after a neuter stem with *śi*. Two rules modify the verbal stem-forming affix: By 3.1.44 *cli* is replaced by the *s*-aorist stem-forming affix *sic*, and 2.4.77 deletes it after the root *dā* before a verbal termination termed *parasmaipada* by 1.4.99. Four rules modify the nominal and verbal stems: 7.1.72 provides the augment *n* after the final vowel of the vowel-final neuter stem *phala* before the termination *śi* which is termed *sarvanāmasthāna* by 1.1.42; 6.4.8 lengthens the penultimate vowel of an *n*-final stem before such a termination, 7.3.102 lengthens the final vowel of an *a*-final stem before a nominal termination that begins with a semivowel, nasal, *jh*, or *bh* (here the *y* in *ya*), and 6.4.71 adds the augment *a* to the beginning of a stem followed by a termination that replaces *luṅ*, *laṅ*, or *lṛṅ*. Finally, a purely phonetic rule applies: In *phalāni*, 6.1.77 replaces the vowel *i* followed by a vowel with *y*.[3]

2 Kārakas

As early as 1964, R. Rocher (1964: 51) criticized the characterization of kārakas as syntactic categories, instead arguing that they are semantic. Calling them syntactico-semantic, Cardona (1976: 215-224) countered that it is suitable to consider kārakas as a level between the purely semantic level and the level at which nominal terminations are introduced (the abstract morphological level in Kiparsky 2002) because the rules that introduce kāraka terms include both semantic and co-occurrence conditions.

It is certainly the case that co-occurrence conditions enter into kāraka classification rules, and therefore that the kāraka classification is an intermediate stage of derivation between that of semantic conditions and that of the introduction of nominal terminations. The intermediate stage is a way of achieving a complex mapping between meaning and speech. It is possible that such an intermediate stage serves merely the purpose of procedural economy and does not imply that kāraka classification constitutes a level in any psychological or structural sense. Pāṇini may conceive of just two levels: semantic (*artha*) and phonetic (*śabda*).

3 *L*-affixes

In their description of levels, Kiparsky and Staal place *l*-affixes at the same level as kārakas. Kiparsky (2002: 3) describes "Assignment of kārakas (th-roles) and of abstract tense" as the function of the first set of rules mapping the semantic level to the morphosyntactic level. The treatment of *l*-affixes by Pāṇini, however, differs markedly from the treatment of kārakas. Kārakas are semantic objects classified by being designated by terms (*sañjñā*). Section 1.4 classifies semantic objects intended to be expressed by a speaker in relational categories by calling them by a kāraka term.

[3] Notes 1-2 apply to Figure 3 as well.

Speech forms are subsequently introduced under the condition that an item designated by a kāraka term is to be denoted. *L*-affixes, in contrast, are introduced under semantic and syntactic conditions, just as other affixes are, and then are replaced by morphological elements; they serve therefore as abstract morphological elements themselves (level 3) rather than as morphosyntactic representations (level 2).[4] Kiparsky's placement of *l*-affixes in level 2 rather than level 3 therefore sharply contrasts with Pāṇini's treatment.

Part of the motivation for assigning *l*-affixes to the level of morphosyntactic representation and their replacements *tip*, *tas*, *jhi*, etc. to the level of abstract morphological representation is to place the basic set of verbal terminations and the basic set of nominal terminations at the same level in the hierarchy and thereby to achieve parallelism between them. 1.4.14 *suptiṅantaṁ padam* refers to basic verbal (*tiṅ*) and nominal (*sup*) terminations alike as the items ending in which a phonetic string is termed a word (*pada*). Just as the basic nominal terminations *su*, *au*, *jas*, etc. are distributed over semantic and syntactic conditions including kāraka and number, the basic verbal terminations *tip*, *tas*, *jhi*, etc. are distributed over the same conditions kāraka and number, and similar conditions such as person (*puruṣa*). Kiparsky (2002: 3) calls the rules that achieve this distribution 'morphological spellout rules'. 3.4.78 *tiptasjhi...* introduces the basic set of verbal terminations just as 4.1.2 *svaujas...* introduces the basic set of nominal terminations. These sutras are read in conjunction with restrictive rules (*niyama*) that achieve the proper distribution over the conditions of number (1.4.21-22),[5] person (1.4.105-108),[6] and kāraka (pāda 2.3 for nominal terminations, and 1.3.13-93 for verbal terminations).

However, the parallelism is incomplete. The verbal terminations introduced by 3.4.78 are not distributed over the conditions of time and mood as the nominal terminations introduced by 4.1.2 are distributed over kārakas. On the contrary, it is rather the *l*-affixes introduced by 3.2.110 *luṅ*, 3.2.111 *anadyatane laṅ*, etc. that are distributed over time and mood. Moreover, the conditions under which *l*-affixes are introduced include kārakas. 3.4.69 *laḥ karmaṇi ca bhāve cākarmakebhyaḥ* provides that *l*-affixes occur under the condition that a *kartṛ* is to be denoted or either a karman or bhāva. The later alternative depends upon whether the root after which the *l*-affix occurs is transitive or intransitive, i.e. occurs with (*sakarmaka*) or without (*akarmaka*) a direct object (*karman*); after intransitive verbal roots the *l*-affix is introduced under the condition that the action itself (*bhāva*) is to be denoted, while after transitive verbal roots the *l*-affix is introduced under the condition that the direct object is to be denoted. 3.4.69 thus accounts for the distribution of *l*-affixes over certain kāraka conditions. In the derivations in Figure 2 and Figure 3, 3.4.69 is clearly out of place; as a rule that maps an abstract morphological element onto a kāraka, it is alone in the section of rules that map from level 1 to level 2. The other rules that map onto kārakas (1.3.78, 3.1.68, and 2.3.2 in Fig. 2; 1.3.78, 2.3.2, and 2.3.13 in Fig. 3) all occur between levels 2 and 3. Verbal terminations, including the so-called basic verbal terminations, are morphophonemic replacements of the *l*-affixes. On the grounds of the

---

[4] Cardona (1997: 496) calls them "abstract affixes".

[5] 1.4.21 *bahuṣu bahuvacanam.* 1.4.22 *dvyekayor dvivacanaikavacane.*

[6] 1.4.105 *yuṣmady upapade samānādhikaraṇe sthāniny api madhyamaḥ.* 1.4.106 *prahāse ca manyopapade manyater uttama ekavac ca.* 1.4.107 *asmady uttamaḥ.* 1.4.108 *śeṣe prathamaḥ.*

parallelism between *l*-affixes and basic nominal terminations, in addition to the fact that they, like the basic nominal terminations *su*, *au*, *jas*, etc. are initially introduced items rather than replacements, *l*-affixes, rather than the basic verbal terminations *tip*, *tas*, *jhi*, etc., would properly be placed at the same level as basic nominal terminations in a fourfold hierarchy of levels.

Moving *l*-affixation to the level of abstract morphological representation would require that basic verbal terminations appear subsequently in the transformation of abstract morphology to phonological output. Such a move is entirely unproblematic. There are no objective criteria to distinguish the level of the basic verbal terminations that replace *l*'s from the level of the nominal terminations that replace the basic nominal terminations *su*, *au*, *jas*, etc. Just as *l*'s are the primary elements introduced after verbal stems, basic nominal terminations *su*, *au*, *jas*, ..., *ṅi*, *os*, *sup* are the primary elements introduced after nominal stems and feminine affixes. Basic verbal terminations replace *l*'s by 3.4.78, and other verbal terminations replace basic verbal terminations by 3.4.79-112, 7.1.3-5, 7.1.35, 7.1.40-46, etc.[7] Replacements include partial as well as total replacements. For example, by 3.4.79 *ṭita ātmanepadānāṁ ṭer e*, under the condition that the basic verbal terminations are marked with *ṭ*, the segment of the basic ātmanepada terminations *ta*, *ātām*, *jha*, etc. that consists of the last vowel and any following consonants is replaced by *e;* while by 3.4.79 the entire basic verbal termination *thās* is replaced by *se*. The basic verbal terminations inherit markers and other properties from the *l* they replace in accordance with the principle, stated in 1.1.56 *sthānivad ādeśo 'nalvidhau*, that replacements have the status of their substituends. Having the status of their substituends likewise extends to the replacements of basic verbal terminations so that verbal forms qualify to be termed *pada* by 1.4.14.

There is no segregation of the type of conditions under which replacements of basic verbal terminations and their subsequent replacements occur, nor any segregation of such conditions according to the location of the sūtras that provide such replacements in the *Aṣṭādhyāyī*. Replacements of the basic terminations in the third adhyāya include phonological conditions, and subsequent replacements in the seventh adhyāya include semantic conditions. For example, 3.4.109-111 include morphological and phonological conditions in the provision that *jus* replaces the *jhi* that replaces *l* marked with *ṅ*. Thus 3.4.109 *sijabhyastavidibhyaś ca* includes the condition that the *jhi* follows the vikaraṇa *sic*, a reduplicated root (*abhyasta*), or the class 2 root *vid;* 3.4.110 *ātaḥ* includes the phonological condition that the *jhi* follows an *ā*-final root after the deletion (*luk*) of *sic;*[8] and 3.4.111 *laṅaḥ śākaṭāyanasyaiva* allows the replacement, in the opinion of Śākaṭāyana, also if the *jhi* that replaces *laṅ* follows an *ā*-final root. On the other hand, 7.1.35 *tuhyos tātaṅ āśiṣy anyatarasyām* provides that *tātaṅ* optionally replaces *tu* or *hi*, which are themselves derived from the basic verbal terminations *tip* and *sip* respectively by 3.4.86-87, under the semantic condition that a wish is to be expressed (*āśiṣi*).

---

[7] Cardona (1997: 487-496) analyses the abstraction of a set of basic verbal terminations first introduced as replacements of *l* by 3.4.78 from verbal terminations that occur in various tenses, aspects, and moods and (1997: 273-279) discusses rules that derive the occuring verbal terminations from the basic verbal terminations.

[8] Cardona (1997: 278) provides details of the derivation of examples.

Likewise, basic nominal terminations are replaced under phonological conditions as well as semantic conditions. For example of the former, after *a*-final stems 7.1.9 *ato bhisa ais* replaces the basic nominal termination *bhis* by *ais*, 7.1.12 *ṭāṅasiṅasām ināṭsyāḥ* replaces the basic nominal terminations *ṭā*, *ṅasi*, and *ṅas* by *ina*, *at*, and *sya*, and 7.1.13 replaces the basic nominal termination *ṅe* by *ya*. For example of the latter, 7.1.19 *napuṁsakāc ca* and 7.1.20 *jaśśasoḥ śiḥ* replace the basic dual and plural first-triplet nominal terminations by *śī* and *śi* respectively after neuter stems.

The fact that there are no objective criteria to distinguish the character of replacements of *l*-affixes from replacements of nominal terminations makes the relocation of basic verbal terminations to the chain of morphophonemic changes that occur in the transformation of abstract morphology to phonological output entirely suitable.

4 Abstract Morphology Versus Phonology

The claim that the phonological output form resides on a different level from the abstract morphological representation is problematic. The abstract morphological representation often appears unchanged as the final phonological output, without having been subject to any additional rule. In the example *devadatta odanaṁ pacati* discussed in section I above (Fig. 2), the affix *-ti* in *pacati*, remains unchanged except for the dropping of the marker *p*.

Conversely, the abstract morphological representation often undergoes more than one permutation before arriving at its final phonological output form. The number of permutations is not correlated with the number of levels. In the same example, the final *s* in *devadattas* (*devadatta+su*) is placed at the level of abstract morphological representation (level 3). The *s* is first changed to *y* by 8.3.17 and then to zero (*lopa*) by 8.3.19 undergoing replacement twice in stepping one level. Figure 3 shows several instances in which there are multiple stages of derivation that take place in transforming abstract morphology to phonological output. Most notably 3.1.44 replaces *cli* (introduced at the level of abstract morphology by 3.1.43) with *sic* which 2.4.77 subsequently deletes.

In contrast to *pacati* in Figure 1, an extra stage of replacement occurs in the derivation of the form *pacanti* (3pa pre: *pac-a-anti* < *pac-a-jhi* < *pac-jhi* < *pac-laṭ*). The *l* of *laṭ* is replaced by *jhi* in accordance with 3.4.78 *tiptasjhi...* and then the cover symbol *jh* is replaced by *ant* after *a*-final stems in accordance with 7.1.3 *jho 'ntaḥ*. The symbol *jh* is replaced by *at* instead after reduplicated stems in accordance with 7.1.4 *ad abhyastāt* and after stems that do not end in *a* before ātmanepada terminations in accordance with 7.1.5 *ātmanepadeṣv anataḥ*. Thus are accounted for forms such as *dadati* (3pa pre *dā* 'give') and *cinvate* (3pm pre *ci* 'collect') respectively. The use of the cover symbol *jh* achieves a valuable generalization in unifying the verbal terminations of the third person plural that do and do not contain *n*. Without privileging either *ant* or *at* as the more basic termination, the former of which is more common in parasmaipada terminations and the latter of which is more common in ātmanepada terminations, positing *jh* as basic nevertheless achieves the same economy of rules as would be achieved by positing *ant* as the basic termination.

It is certainly arguable that in some instances the choice of abstract morphological representation, whether it ever appears in phonological output or not, is motivated by procedural economy and proportional representation of forms that actually occur. Cardona (1997: 330-332) discusses cover symbols and (490-492) demonstrates the economy and elegance of the inclusion of the cover symbol *jh* in the basic verbal terminations. The reasons for the use of the abstract symbol *cli* as the basic aorist stem-forming-affix are less apparent. To what extent procedural economy and proportional representation in phonological output serve as the criteria to determine the choice of abstract morphological representation requires further investigation. It is nevertheless certainly clear that the choice of the particular abstract morphological representation in some cases is identical to a final phonological output; in other cases it requires several stages of transformation to reach phonological output; and in still others it never appears as phonological output. The last is precisely what the previous section argued is the situation with *l*-affixes. Just as *jh* and *cli* are abstract morphological representations at level 3, *l*, with various markers, is the abstract morphological representation of all verbal terminations. Since the number of permutations is not correlated with the number of levels, the fact that *l*'s undergo more than one permutation before reaching final phonological output form in most verb forms is not grounds for segregating these permutations into separate levels, just as it is not grounds for positing a separate level for the *y* posited as a replacement for the nominal termination *su* in accordance with 8.3.17 (Fig. 2), or for *sic* which replaces *cli* by 2.4.77 (Fig. 3), both of which undergo an additional permutation before appearing in final phonological output.

Once *l*-affixes are postponed one level to the level of abstract morphology, basic verbal terminations *tip*, *tas*, *jhi*, etc. are seen to be simply one additional morphophonemic modification of *l*-affixes, just like, for example, the imperative terminations *tu*, *tām*, *antu*, etc. which are further morphophonemic modifications of the basic verbal terminations *tip*, *tas*, *jhi*, etc., and just like *ais* (introduced after *a*-final stems by 7.1.9 *ato bhisa ais*) which is a morphophonemic modification of the basic nominal termination *bhis*.

The only justification for considering that *l*-affixes belong to the level of morphosyntactic representation rather than to the level of abstract morphological representation like other abstract affixes such as *cli* and *jh* is that the conditions for the replacement of *l*-affixes include semantics and syntax while the conditions for the replacement of *cli* (by 3.1.44) and *jh* (by 7.1.3-5) are only morphological and phonological. However, this criterion is invalid. As Scharf (2008: sections IVB and IVD2) pointed out, Houben (1999) demonstrated that semantic factors directly serve as conditions even in phonological rules, and Cardona (personal communication) pointed out that most phonological rules include syntactic conditions. Houben (1999: 46) illustrated the direct use of semantic and pragmatic factors as conditions for phonetic modifications to strings in the section of rules 8.2.82-108. Factors such as giving a responding greeting to someone belonging to a higher caste than a śūdra (8.2.83 *pratyabhivāde 'śūdre*) and calling from a distance (8.2.84 *dūrād dhute*) conjoin with the syntactic condition, specified in the heading to the section (8.2.82 *vākyasya ṭeḥ pluta udātta*), that the string be a sentence to condition vowel prolongation and high tone. Since semantic and syntactic conditions can serve as conditions in rules that map from abstract morphological representation (level 3) to phonological output form (level 4) it is not the case that conditions are restricted to the levels from which and to

which they map. Kiparsky (2002) conceded that rules incorporate conditions at prior levels as well. Therefore the fact that rules that replace *l*-affixes include semantic and syntactic conditions is not sufficient grounds for preponing *l*-affixes to the level of morphosyntactic representation. The real motivation for doing so must be recognized as a twentieth century conception of a fourfold distinction between semantics, syntax, morphology, and phonetics.

5 Conclusions

Stages of replacement vary greatly in the production of speech forms; there is no clear association between those stages and any psychological or conceptual level. In distinction to potentially multiple stages of affixes and their replacements, it seems to me that just one level is involved once an affix has been introduced. The fact that Pāṇini uses the technique of replacement for the derivation of the final output form from an abstract morphological representation indicates that the replacement is considered to belong to the same level rather than to a different one; it belongs to the morphophonemic level as opposed to the syntacticosemantic level.

The semantic and syntactic levels are properly coalesced in a syntacticosemantic level and the abstract morphological and the morphophonemic levels are properly coalesced in a single morphophonemic level. While Pāṇini derives forms through numerous un-correlated stages of derivation, he makes a clear distinction between the level of meaning and the level of speech.

The concept of levels in Pāṇinian grammar, and the hierarchy of four levels proposed by Kiparsky and Staal, was inspired by divisions that evolved in modern linguistics. It is anachronistic to read them into the *Aṣṭādhyāyī.* Kiparsky himself (2002: 2) hedges his attribution of levels to Pāṇini calling them, "what we (from a somewhat anachronistic modern perspective) could see as different levels of representation." Pāṇini's grammar certainly worked with two levels: meaning and speech. Its derivational procedure certainly included more than two stages. However, it appears forced to press the derivational stages into a conceptual hierarchy of levels between the purely semantic and the purely phonetic, particularly into a four-level hierarchy corresponding to modern linguistic divisions.[9] Consequently, it would be inappropriate to call a computational implementation of such a four-level hierarchy a close model of Pāṇinian methodology.

In working within the two levels meaning and speech, Pāṇini does stratify these levels so that it is possible to consider that there are four levels, though these do not align neatly with the modern conceptions of semantics, syntax, morphology, and phonology. The level of meaning can be stratified into an initial stage of naive worldly semantics as opposed to a subsequent stage of syntacticosemantic organization ready to serve as conditions for morphophonemic rules. The level of sound can be stratified into an initial stage in which basic morphophonemic elements, including abstract

[9] Hyman (2003: 188-89) argues that Herodian's recognition of three types of linguistic errors--namely, barbarism, solecism, and acyrologia--corresponds to the threefold distinction of phonology, morphosyntax, and semantics.

morphological elements, are introduced and a final stage of the finished phonological form. In this way one does arrive at a four-fold hierarchy with three types of rules: rules that organize the syntacticosemantic level, rules that introduce basic elements, and rules that modify introduced elements. Rules that organize the syntacticosemantic field include kāraka classification. Rules that introduce basic elements include the rules that introduce affixes after roots and stems in chapters 3-5 of the *Aṣṭādhyāyī*. Rules that modify introduced elements include rules of augmentation, substitution, and deletion. The criteria for the segregation of such rules are obvious in the syntax and purport of the rules themselves.

6 Implications for Computational Modeling

Because it is incorrect to assert that *l*-affixes, which would be more appropriately placed in the level of abstract morphological representation, and kārakas, which belong to the level of morphosyntactic representation, occupy the same level in a four-level hierarchy, therefore a four-module implementation based on such a hierarchy would not produce a close computational model of Pāṇinian procedure if it implemented rules that provide *l*-affixes in the same module as rules that classify kārakas. Likewise, because it is incorrect to assert that verbal terminations, which are morphophonemic modifications of *l*'s brought about by 3.4.78, etc., and the nominal terminations *su*, *au*, *jas*, etc., which are affixes that serve as abstract morphological representation initially introduced by 4.1.2, occupy the same level, it would not produce a close computational model of Pāṇinian procedure to implement these rules in the same module. In a computational model based upon a hierarchy of levels that modeled Pāṇinian procedure, *l*-affixes would have to be introduced in the same module that introduced other affixes, in a module prior to a module that provided morphophonemic replacements of them, and in a module subsequent to one that classified kārakas. Verbal terminations would have to replace *l*-affixes in the same module that provided other morphophonemic replacements of abstract morphological representations, and in a module subsequent to one that initially introduced affixes.

References

Cardona, G.: Pāṇini: A Survey of Research. Mouton, The Hague (1976)

Chomsky, N.: Syntactic Structures. Mouton, The Hague (1957)

Houben, J.: 'Meaning statements' in Pāṇini's grammar: on the purpose and context of the *Aṣṭādhyāyī*. Studien zur Indologie und Iranistik 22, 23–54 (1999) [2001]

Hyman, M.: One word solecisms and the limits of syntax. In: Swiggers, P., Wouters, A. (eds.) Syntax in Antiquity. Orbis Supplementa 23. Monographs published by the International Center of General Dialectology, Louvain, pp. 179–192. Peeters, Leuven (2003)

Kiparsky, P.: On the Architecture of Pāṇini's Grammar. Paul Kiparsky's Home Page (2002), `http://www.stanford.edu/~kiparsky/` (See Kiparsky, 2008)

Kiparsky, P.: On the Architecture of Pāṇini's Grammar. In: Huet, G., Scharf, P. (eds.) Topics in Sanskrit Computational Linguistics. LNCS. Springer, Heidelberg (2008) (See Kiparsky, 2008)

Kiparsky, P., Staal, J.F.: Syntactic and semantic relations in Pāṇini. Foundations of Language 5, 83–117 (1969)

Rocher, R.: ‘Agent’ et ‘objet’ chez Pāṇini. Journal of the American Oriental Society 84, 44–54 (1964)

Scharf, P.: Modeling Pāṇinian Grammar. In: Huet, G., Kulkarni, A. (eds.) Proceedings of First International Symposium on Sanskrit Computational Linguistics, pp. 77–94 (2007), http://sanskrit.inria.fr/Symposium/DOC/Scharf.pdf, http://sanskrit.inria.fr/Symposium/DOC/Scharf_Slides.pdf

Scharf, P.: Modeling Pāṇinian Grammar. In: Huet, G., Kulkarni, A., Scharf, P. (eds.) Topics in Sanskrit Computational Linguistics. LNCS, Springer, Heidelberg (2008)

On the Construction of Śivasūtra-Alphabets

Wiebke Petersen

Heinrich-Heine University Düsseldorf
Universitätsstr. 1, 40225 Düsseldorf, Germany
petersew@uni-duesseldorf.de

Abstract. In the present paper, a formalization of the technique used by Pāṇini in his Śivasūtras for the denotation of sound classes is given. Furthermore, a general notion of Śivasūtra-alphabets and of Śivasūtra-sortability is developed. The presented main theorem poses three sufficient conditions for the Śivasūtra-sortability of sets of classes. Finally, the problem of ordering sets of classes which are not Śivasūtra-sortable is tackled and an outlook on modern problems which could be approached by Pāṇini's technique is given.

Keywords: Śivasūtras, Pāṇini, orders, lattices.

1 Introduction

1.1 Pāṇini's Śivasūtra-Technique

Among linguists Pāṇini's grammar of Sanskrit is acknowledged to be the culmination point of ancient Indian grammar:

> Indian linguistics originated among reciters who wanted to preserve their Vedic heritage and apply it in ritual. Unconcerned with meaning, they concentrated on form and incorporated a good measure of linguistic analysis that culminated in the Sanskrit grammar of Pāṇini. (Staal, 2006)

Although more than 2 000 years old, Pāṇini's grammar is rather accurately preserved due to the fact that it was soon considered to be the standard grammar of Sanskrit. Thereby the originally descriptive grammar of a living language achieved the status of a normative, prescriptive grammar (cf. Staal, 1995). Situated in the oral culture of ancient India, Pāṇini's grammar was designed for repetitive recitation. Thus the grammar is necessarily presented in a purely linear form, and its compactness was particularly desirable.

Its main part consists of about 4 000 rules, many of them phonological rules which describe the complex system of Sanskrit Sandhi (cf. Böhtlingk, 1887). Phonological rules are typically of the form "sounds of class A are replaced by sounds of class B if they are preceded by sounds of class C and followed by sounds of class D", which in modern phonology is usually denoted as

$$A \rightarrow B/_{C__D} \ . \qquad (1)$$

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 78–97, 2009.

अइउण् ॠऌक् एओङ् ऐऔच् हयवरट् लण् ञमङणनम् झभञ्
घढधष् जबगडदश् खफछठथचटतव् कपय् शषसर् हल् (I)

a·i·uṇ ṛ·ḷk e·oṅ ai·auc hayavaraṭ laṇ ñamaṅaṇanam jhabhañ
ghaḍhadhaṣ jabagaḍadaś khaphachaṭhathacaṭatav kapay śaṣasar hal (II)

Fig. 1. Pāṇini's Śivasūtras in linear form (I: Devanāgarī script; II: Latin transcription)

Since Pāṇini's grammar has been designed for oral tradition, it makes no use of visual symbols (like arrows, slashes ...) to indicate the role of the sound classes in a rule. Instead, Pāṇini takes natural case suffixes which he uses meta-linguistically in a formalized way in order to mark the role a class plays in a rule. In Pāṇinian style rule (1) becomes

$$A + \text{genitive}, B + \text{nominative}, C + \text{ablative}, D + \text{locative}\ . \tag{2}$$

Since constantly repeating the single sounds of each class involved in a rule is not economical, an appropriate phonological description must involve a method to denote the sound classes. The method should be such that it is easier to address a natural phonological class than an arbitrary set of sounds (cf. Kornai, 1993, 2008). A wide-spread technique in modern phonology is to build up a structured system of phonetic features (e.g., [±consonantal] or [±voiced]) in order to define the phonologically relevant sound classes. The aim is to identify phonetic features, i.e., features that are motivated by properties of the isolated sounds, by which the sounds can be classified into phonological classes, i.e., into classes of sounds with analogous behavior in the same speech contexts. Unfortunately, this leads to the problem of choosing and naming features and often involves the danger of defining ad-hoc features.

Pāṇini's technique for the denotation of sound classes allows him to do completely without features. His grammar of Sanskrit begins with 14 sūtras, the so-called *Śivasūtras*, which are quoted in Fig. 1 in their original linear form and in Fig. 2 in the tabular form given in Kiparsky (1991). Each single sūtra consists of a sequence of sounds which ends in a consonant, the so-called *anubandha*. This last consonant of each sūtra is used meta-linguistically as a marker to indicate the end of a sūtra. According to Misra (1966) the system behind the choice of the consonants used as anubandhas is unknown. Together the Śivasūtras define a linear list of Sanskrit sounds which is interrupted by marker elements (anubandhas). In his grammar Pāṇini uses *pratyāhāras*, i.e., pairs consisting of a sound and an anubandha in order to designate the sound classes on which a rule operates. Such a pair denotes the sounds in the interval between the sound and the anubandha; e.g., the pratyāhāra iC denotes the class {i, u, ṛ, ḷ} as depicted in Fig. 3.[1] Pratyāhāras are often used in rules of type (2) where they replace the open place-holders A, B, C and D.

[1] To simplify matters we ignore here that a pratyāhāra actually denotes the ordered list of sounds in the interval and not just the unordered class of sounds .

1.	a	i	u			Ṇ
2.				ṛ	ḷ	K
3.		e	o			Ṅ
4.		ai	au			C
5.	h	y	v	r		Ṭ
6.					l	Ṇ
7.	ñ	m	ṅ	ṇ	n	M
8.	jh	bh				Ñ
9.			gh	ḍh	dh	Ṣ
10.	j	b	g	ḍ	d	Ś
11.	kh	ph	ch	ṭh	th	
			c	ṭ	t	V
12.	k	p				Y
13.			ś	ṣ	s	R
14.	h					L

Fig. 2. Pāṇini's Śivasūtras in tabular form (the default, syllable-building vowel *a* is left out)

1.	a	ⓘ	**u**			Ṇ
2.				**ṛ**	**ḷ**	K
3.		**e**	**o**			Ṅ
4.		**ai**	**au**			**C**
5.	h	y	v	r		Ṭ

Fig. 3. Example of a pratyāhāra: iC = {i,u,ṛ,ḷ,e,o,ai,au}

There is a longstanding debate on how Pāṇini developed the Śivasūtras and whether he arranged the sounds in the best way possible (cf. Böhtlingk, 1887; Faddegon, 1929; Staal, 1962; Kiparsky, 1991). Note that exactly one sound, namely *h*, occurs twice in the list of the Śivasūtras (in the 5th and in the 14th sūtra). Nowadays it is generally assumed that the order of the sounds in the Śivasūtras is primarily determined by the structural behavior of the sounds in the rules of Pāṇini's grammar and that the arrangement of the sounds is chosen such that economy or rather brevity is maximized (cf. Staal, 1962; Misra, 1966; Cardona, 1969; Kiparsky, 1991). Kiparsky (1991) argues "that the structure of the Śivasūtras is entirely explicable on systematic grounds [. . . and] that no other principles are needed than those used in the construction of the rest of Pāṇini's grammar, namely the principle of economy and the logic of the special case and the general case."

In Petersen (2004) it has been formally proven that there is no shorter solution than the Śivasūtras to the problem of ordering the sounds of Sanskrit in a by markers interrupted, linear list with as few repeated sounds as possible such that each phonological class which is denoted by a sound-marker pair (i.e., a pratyāhāra) in Pāṇini's grammar can be represented by such a pair with respect to the list. Hence, Pāṇini was forced to duplicate one sound, namely *h*, in the Śivasūtras and he used a minimal number of markers. Actually, it can be shown

that there are nearly 12 000 000 alternative sound lists interrupted by markers which fulfill the above mentioned conditions and which are of the same length as the Śivasūtras (Petersen, 2008). The question whether the actual list chosen by Pāṇini in the Śivasūtras results, as Kiparsky (1991) argues, from the 'principle of economy' and the 'logic of the special case and the general case' and not from the 'principle of historic continuity' (Cardona, 1969) or the 'principle of homorganic continuity' (Staal, 1962) cannot be answered by the mathematical reasoning in Petersen (2008).

The present paper focuses not so much on the concrete list of the Śivasūtras as former ones (cf. Petersen, 2004, 2005), but concentrates more on the general technique of ordering entities in a list which is interrupted by marker elements such that each class of entities out of a given set of classes forms an interval and thus can be unambiguously addressed by a pair consisting of an entity and a marker. In particular it is examined under which conditions it is possible to construct such a list without being forced to include an entity twice.

1.2 General Problem of S-sortability

As a start we will simplify Pāṇini's Śivasūtra-technique by abandoning the claim that the target list is interrupted by markers and that each class which is denotable with respect to the list forms an interval which ends immediately before a marker. Thus the simplified problem states as follows:

Problem 1. Given a set of classes, order the elements of the classes in a linear order such that each single class forms a continuous interval with respect to that order.

The target orders will be called S-orders:

Definition 1. *Given a finite base set $\mathcal{A}$ and a set of subsets Φ with $\bigcup \Phi = \mathcal{A}$, a linear order $<$ on $\mathcal{A}$ is called a* Śivasūtra-order *(or short* S-order*) of $(\mathcal{A}, \Phi)$ if and only if the elements of each set $\phi \in \Phi$ form an interval in $(\mathcal{A}, <)$, i.e., $\forall \phi \in \Phi$: if ϕ_{min} is the minimum of ϕ w.r.t. $(\mathcal{A}, <)$ and ϕ_{max} is the maximum of ϕ, then there is no $a \in \mathcal{A} \setminus \phi$ s.th. $\phi_{min} < a < \phi_{max}$.*

Furthermore, $(\mathcal{A}, \Phi)$ is said to be S-sortable *if and only if there exists an S-order $(\mathcal{A}, <)$ of $(\mathcal{A}, \Phi)$.*

Example 1. Given the base set $\mathcal{A} = \{a, b, c, d, e, f, g, h, i\}$ and the set of classes $\Phi = \{\{d, e\}, \{a, b\}, \{b, c, d, f, g, h, i\}, \{f, i\}, \{c, d, e, f, g, h, i\}, \{g, h\}\}$, $(\mathcal{A}, \Phi)$ is S-sortable and $a \prec b \prec c \prec g \prec h \prec f \prec i \prec d \prec e$ is an S-order of $(\mathcal{A}, \Phi)$.[2]

It is important to note that not all sets of classes are S-sortable. For instance, since the duplication of at least one sound element in the Śivasūtras is unavoidable, the set of classes defined by the sound classes in Pāṇini's grammar which are denoted by pratyāhāras is not S-sortable. Orders, like the one underlying the

[2] As usual, $\prec$ stands for the binary predecessor relation, i.e., $a \prec b$ if and only if $a < b$ and there is no c such that $a < c < b$.

Śivasūtras, which contain at least one element twice will be called S-orders *with duplications*. A smaller example of a non S-sortable set of classes is given here:

Example 2. Given the base set $\mathcal{A} = \{a, b, c, d, e, f\}$ and the set of classes $\Phi = \{\{d, e\}, \{a, b\}, \{b, c, d\}, \{b, c, d, f\}\}$, $(\mathcal{A}, \Phi)$ is not S-sortable (without duplications).

One of the major aims of the present paper is to examine the conditions which S-sortable sets of classes fulfill and to show how these conditions can be constructively applied to different tasks: (1) the building of concrete S-orders, (2) the identification of best candidates for duplication in the case of non S-sortable sets of classes, (3) the insertion of a minimal amount of marker elements such that each class forms an interval that ends immediately before a marker. S-orders which are interrupted by marker elements are called S-alphabets and defined as follows:

Definition 2. *Given a finite base set $\mathcal{A}$ and a set of subsets Φ with $\bigcup \Phi = \mathcal{A}$, a* Śivasūtra-alphabet *(short* S-alphabet*) of $(\mathcal{A}, \Phi)$ is a triple $(\mathcal{A}, \Sigma, <)$ with*

- *Σ is a finite set of markers with $\mathcal{A} \cap \Sigma = \emptyset$,*
- *$<$ is a linear order on $\mathcal{A} \cup \Sigma$*

if and only if for each $\phi \in \Phi$ there exists $a \in \phi$ and $M \in \Sigma$ such that $\phi = \{b \in \mathcal{A} \,|\, a \leq b < M\}$ (aM is called the pratyāhāra or S-encoding *of ϕ).*

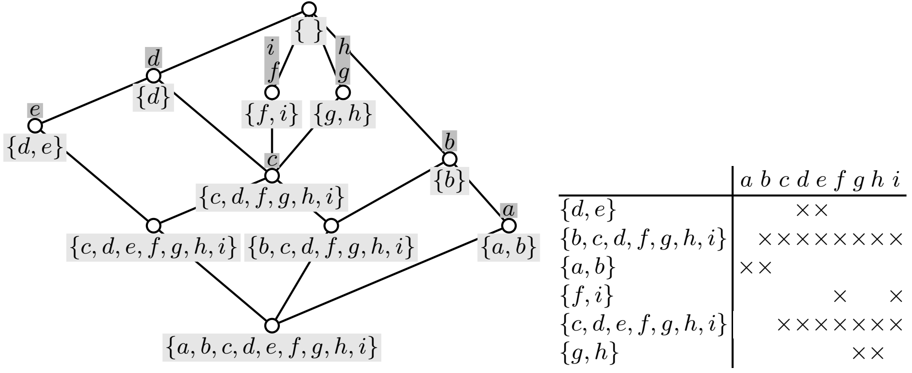
Furthermore, $(\mathcal{A}, \Phi)$ is said to be S-encodable *if and only if there exists an S-alphabet $(\mathcal{A}, \Sigma, <)$ of $(\mathcal{A}, \Phi)$.*

It follows from definition 2 that whenever $(\mathcal{A}, \Sigma, <)$ is an S-alphabet of $(\mathcal{A}, \Phi)$ then $(\mathcal{A}, < |_{\mathcal{A}})$ is an S-order of $(\mathcal{A}, \Phi)$. Furthermore, since every S-order can be trivially enhanced into an S-order by inserting a marker behind each element, it is true that each S-sortable set of classes is S-encodable and vice versa.

2 Main Theorem on S-sortability

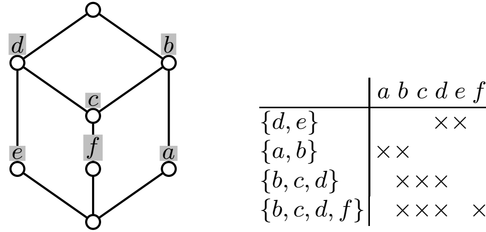
The main theorem on S-sortability depends on two constructs taken from Formal Concept Analysis (FCA), which is a mathematical theory for the analysis of data (cf. Ganter and Wille, 1999). For our purposes, we do not need to evolve the whole apparatus of FCA, it is sufficient to define what we understand by the formal context and the concept lattice of a set of classes $(\mathcal{A}, \Phi)$: Given a base set $\mathcal{A}$ and a set of subsets Φ, the *formal context* of $(\mathcal{A}, \Phi)$ (or the $(\mathcal{A}, \Phi)$-context) is the triple $(\Phi, \mathcal{A}, \ni)$ and the concept lattice of $(\mathcal{A}, \Phi)$ (or the $(\mathcal{A}, \Phi)$-lattice) is the ordered set $(\mathcal{A} \cup \{\psi \,|\, \psi = \bigcap \Psi \text{ with } \Psi \subseteq \Phi\}, \supseteq)$.

Given the base set $\mathcal{A}$ and the set of classes Φ in example 1, the formal context of $(\mathcal{A}, \Phi)$ and the Hasse-diagram of the concept lattice of $(\mathcal{A}, \Phi)$ are depicted in Fig. 4. The formal context is given in form of a cross table as usual. Its concept lattice is constructed as follows: All elements of Φ and all possible intersections of elements of Φ are ordered by the set-inclusion relation such that subsets are



	a	b	c	d	e	f	g	h	i
$\{d, e\}$				×	×				
$\{b, c, d, f, g, h, i\}$		×	×	×	×	×	×	×	×
$\{a, b\}$	×	×							
$\{f, i\}$						×			×
$\{c, d, e, f, g, h, i\}$			×	×	×	×	×	×	×
$\{g, h\}$							×	×	

Fig. 4. Concept lattice (left) and formal context (right) of $(\mathcal{A}, \Phi)$ in example 1



	a	b	c	d	e	f
$\{d, e\}$				×	×	
$\{a, b\}$	×	×				
$\{b, c, d\}$		×	×	×		
$\{b, c, d, f\}$		×	×	×		×

Fig. 5. Concept lattice (left) and formal context (right) of $(\mathcal{A}, \Phi)$ in example 2

placed above their supersets. The *Hasse-diagram* of an ordered set is the directed graph whose vertices are the elements of the set and whose edges correspond to the upper neighbor relation determined by the order. An ordered set is said to be Hasse-planar if its Hasse-diagram can be drawn without intersecting edges. Hence, the concept lattice in Fig. 4 is Hasse-planar.

The node labeling in Fig. 4 is twofold: The labels below the nodes indicate the corresponding sets. For the labels above the nodes a more economic labeling is chosen which assigns each element of the base set $\mathcal{A}$ to the node corresponding to the smallest set which contains the element. The labels below the nodes are superfluous as they can be reconstructed from the others by collecting all labels attached to nodes which can be reached by moving along paths upwards in the graph. From now on, solely the upper labels will be used in figures of concept lattices, as seen in Fig. 5, which shows the concept lattice for example 2.

The main theorem on S-sortability states three equivalent, sufficient conditions which a set of classes must fulfill in order to be S-sortable. The individual conditions will be explained in detail in the succeeding subsections.

Theorem 1. *A set of classes* $(\mathcal{A}, \Phi)$ *is S-sortable if and only if one of the following equivalent statements is true:*

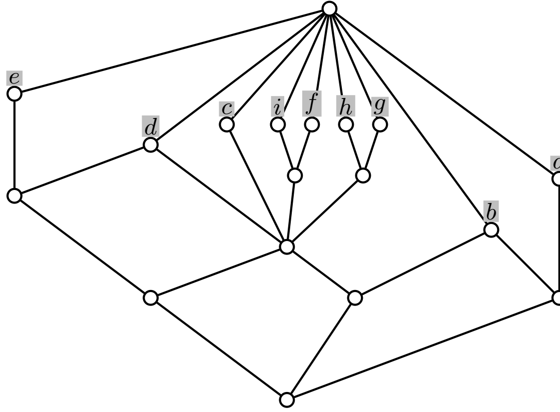


Fig. 6. Enlarged concept lattice for example 1

Condition 1: *Let* $\tilde{\Phi} = \Phi \cup \{\{a\} \mid a \in \mathcal{A}\}$*. The concept lattice of the enlarged set of classes* $(\mathcal{A}, \tilde{\Phi})$ *is Hasse-planar.*

Condition 2: *The concept lattice of* $(\mathcal{A}, \Phi)$ *is Hasse-planar and for any* $a \in \mathcal{A}$ *there is a node labeled a in the S-graph of the concept lattice.*

Condition 3: *The Ferrers-graph of the enlarged* $(\mathcal{A}, \tilde{\Phi})$*-context is bipartite.*

Although all three conditions depend on properties of graphs, they are of different nature. The first one demands that the Hasse-diagram of a concept lattice can be drawn without intersecting edges; the second one relies on the positions of certain labels in such a Hasse-diagram; and the third one depends on the bipartity of so-called Ferrers-graphs. Instead of giving the proof of the theorem in isolation, the following subsections treat the three conditions for S-sortability one by one. For each condition, illustrational examples are given, a proof of its sufficiency is sketched, and it is demonstrated how the condition can be applied in the construction of S-alphabets with as few duplicated elements as possible and a minimal number of markers.

2.1 First Condition for S-sortability: Main Planarity Criterion

Condition 1 relates S-sortability with Hasse-planarity of enlarged concept lattices. Here, a set of classes gets enlarged by adding each element of the base set as a singleton set to the set of classes, e.g., in the case of example 1 the classes $\{a\}$, $\{b\}$, $\{c\}$, ..., $\{i\}$ have to be added. The condition states that a set of classes is S-sortable if and only if the concept lattice of the so enlarged set of classes is Hasse-planar, i.e., if it is possible to draw its Hasse-diagram without intersecting edges. Figure 6 shows a plane drawing of the enlarged concept lattice for the set of classes taken from example 1.

Figure 7 shows the Hasse-diagram of the enlarged concept lattice that belongs to the set of classes in example 2 which is not S-sortable. In the case of this small

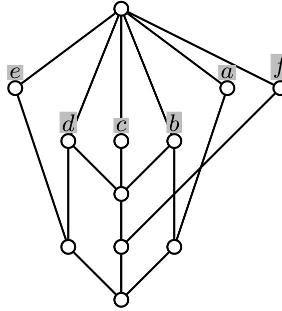


Fig. 7. Enlarged concept lattice for example 2

lattice it can be easily verified that it is impossible to draw its Hasse-diagram without intersecting edges.

Condition 1 is proven in detail in Petersen (2008). The fact that the existence of a plane drawing of the Hasse-diagram of an enlarged concept lattice implies the existence of an S-order follows immediately from the definition of our concept lattices: Since concept lattices order sets by set inclusion it is ensured that in the case of an enlarged set of classes the labels belonging to the elements of one class form an interval in the sequence defined by the left-to-right order of the labels in a plane drawing of the Hasse-diagram of the concept lattice. It can be easily seen that this guarantees that the left-to-right order of the labels in a plane drawing of the Hasse-diagram of a concept lattice of an enlarged set of classes $(\mathcal{A}, \tilde{\Phi})$ defines an S-order of $(\mathcal{A}, \Phi)$. For instance, the S-order defined by the plane Hasse-diagram in Fig. 6 is $e \prec d \prec c \prec i \prec f \prec h \prec g \prec b \prec a$.

The second statement, i.e. that the existence of an S-order implies the existence of a plane drawing of the enlarged Hasse-diagram, was first proven in Petersen (2004). The proof is based on the controlled construction of a drawing of the enlarged concept lattice for each S-order which ensures that the drawing is plane. The resulting drawing is such that the left-to-right order of the labels equals the original S-order.

Since several plane drawings leading to different S-orders usually exist for a concept lattice of a set of classes, our construction method does not deterministically result in one S-order. In fact, the proof of the theorem above implies that for every S-order there exists a plane drawing of the concept lattice from which it can be read off. For instance, for the Hasse-diagram in Fig. 6 one finds 48 plane drawings leading to 48 distinct S-orders of the set of classes taken from example 1.

See Petersen (submitted) for a discussion on why and how S-orders can be fruitfully applied to the problem of ordering books in a library or products in a warehouse. In short, the applicability of S-orders to these problems is based on the fact that in S-orders elements belonging to one class are placed in close distance to each other.

As demonstrated, condition 1 reduces the problem of S-sortability nicely to the Hasse-planarity of certain concept lattices. However, in practice condition 1 is

problematic for two reasons in particular: First, the condition is non-constructive for the problem of inducing S-alphabets with minimal marker sets. If one finds a plane drawing of the Hasse-diagram of an enlarged set of classes, it is always possible to read off an S-order, but usually not an S-alphabet of the original set of classes without superfluous markers. Each S-order can be trivially completed into an S-alphabet by inserting a marker element behind each element in the S-order, but such an S-alphabet will usually contain unnecessarily many markers. The problem is that by enlarging a set of classes the information about which elements do not need to be separated by a marker in an S-alphabet gets lost. Condition 2, which operates on concept lattices that are not enlarged, offers a way out of the dilemma.

Second and even worse, condition 1 does not offer an easily verifiable criterion for the S-sortability of a set of classes. The problem of determining whether a plane drawing of a general graph exists is hard. In section 2.3, which treats condition 3, a sufficient criterion for the Hasse-planarity of concept lattices will be presented which can be algorithmically checked.

2.2 Second Condition for S-sortability: Minimizing the Number of Marker Elements

Condition 2 consists of two parts. It states that a set of classes $(\mathcal{A}, \Phi)$ is S-sortable if and only if the following two conditions are fulfilled:

1. The concept lattice of $(\mathcal{A}, \Phi)$ is Hasse-planar.
2. For any $a \in \mathcal{A}$ there is a node labeled a in the S-graph of the concept lattice of $(\mathcal{A}, \Phi)$.

The second part depends on the notion of S-graphs of concept lattices. S-graphs only exist for Hasse-planar concept lattices since their definition is based on plane drawings of Hasse-diagrams: Given a plane drawing of the Hasse-diagram of an $(\mathcal{A}, \Phi)$-lattice, remove the top node and all adjoined edges if it corresponds to the empty set (if the top node does not correspond to the empty set, do not change the drawing). The resulting drawing defines a plane graph, and the boundary graph of the infinite face of this graph is the S-graph of the $(\mathcal{A}, \Phi)$-lattice. In Petersen (2008) it has been proven that for each S-sortable set of classes there exists exactly one S-graph up to isomorphism. Examples of S-graphs are given in Fig. 8.

The proof of condition 2 is based on the following considerations: According to condition 1 a set of classes $(\mathcal{A}, \Phi)$ is S-sortable if and only if the enlarged $(\tilde{\mathcal{A}}, \Phi)$-lattice is Hasse-planar. Since an S-order of $(\mathcal{A}, \Phi)$ is necessarily an S-order of $(\tilde{\mathcal{A}}, \Phi)$ too, it follows that the S-sortability of a set of classes implies the Hasse-planarity of its concept lattice. Hence, for any S-sortable set of classes a plane drawing of the Hasse-diagram of its concept lattice exists which implies the existence of the S-graph of its concept lattice. However, the Hasse-planarity is only a necessary, but not a sufficient precondition for S-sortability as Fig. 5 demonstrates, which shows a plane drawing of the Hasse-diagram of a concept lattice of a set of classes that is not S-sortable.

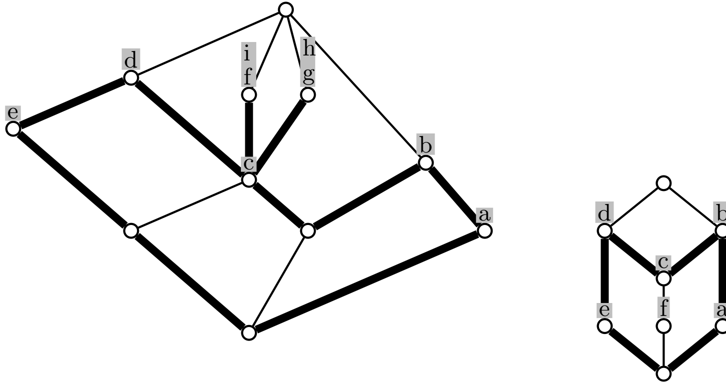


Fig. 8. S-graphs of $(\mathcal{A}, \Phi)$-lattices (left: $(\mathcal{A}, \Phi)$ taken from example 1, right: $(\mathcal{A}, \Phi)$ taken from example 2)

A close investigation of what happens while reducing a plane drawing of an enlarged concept lattice to a plane drawing of the non-enlarged concept lattice will conclude the proof of condition 2: A plane drawing of the non-enlarged concept lattice can be gained from a plane drawing of the enlarged one by contracting all edges leading from lower nodes to nodes which correspond to singleton sets which were added while enlarging the set of classes. For each such node there will be exactly one edge which must be contracted. Hence, the contraction will not destroy the planarity of the graph. Furthermore, a node labeled a (with $a \in \mathcal{A}$) which trivially belongs to the S-graph of the enlarged concept lattice will also belong to the S-graph of the non-enlarged concept lattice. This proves that condition 2 is equivalent to condition 1 and thus that it is a sufficient condition for the S-sortability of a set of classes.

In contrast to condition 1, condition 2 operates immediately on the concept lattice of the original set of classes. Hence, on the concept lattice which involves only those sets for which a pratyāhāra must exist in a corresponding S-alphabet. Therefore, it is possible to develop a procedure for the construction of S-alphabets with minimal marker sets on the basis of the S-graphs treated in condition 2. For a detailed illustration of how the procedure works see Fig. 9 which stepwise illustrates the procedure for the S-sortable set of classes given in example 1.

Procedure for the construction of S-alphabets with minimal marker sets:

1. Start with the empty sequence and choose a walk through the S-graph that:
   - starts and ends at the lowest node,
   - reaches every node of the S-graph,
   - passes each edge not more often than necessary,
   - is oriented such that while moving downwards as few labeled nodes with exactly one upper neighbor as possible are passed.

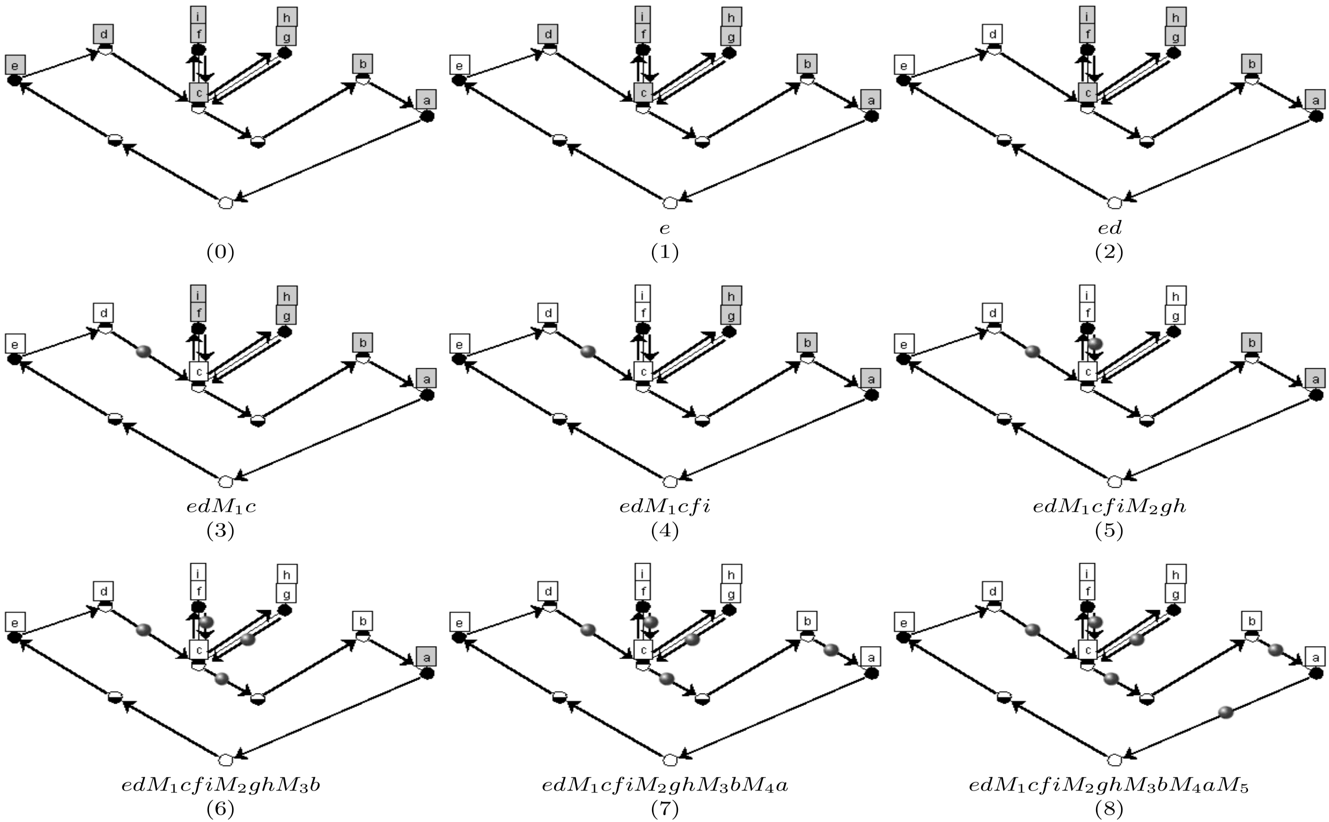


Fig. 9. Sequence of figures to illustrate the procedure for the construction of S-alphabets with minimal marker sets

2. While walking through the S-graph modify the sequence as follows:
   - While moving upwards along an edge do not modify the sequence.
   - While moving downwards along an edge add a new marker to the sequence unless its last element is already a marker.
   - If a labeled node is reached, add the labels in arbitrary order to the sequence, except for those labels which have already been added in an earlier step.

The small example given in Fig. 10 illustrates the importance of choosing a walk through the S-graph that avoids passing labeled nodes while moving downwards. In Petersen (2004) a similar procedure is applied in order to demonstrate that the number of markers in Pāṇini's Śivasūtras cannot be reduced. Note that the procedure is not deterministic, as it usually does not return one single S-alphabet. In Petersen (2008) it has been proven that every S-alphabet with minimal marker set can be derived by this procedure. In the case of Pāṇini's problem the procedure leads to nearly 12 000 000 equally short S-alphabets in which the sound *h* occurs twice.

2.3 Third Condition for S-sortability: Algorithmically Verifiable Criterion

Since it is hard to decide whether a concept lattice is Hasse-planar by examining the concept lattice itself, it is favorable to use a planarity criterion which does not depend on properties of concept lattices (like condition 1 and 2), but on properties of their corresponding formal contexts which can be checked more easily. Condition 3 follows immediately from condition 1 and the following proposition which is proven in Zschalig (2007):

Proposition 1. *The concept lattice of a formal context is Hasse-planar if and only if its Ferrers-graph is bipartite.*

A graph is said to be *bipartite* if it is possible to assign its vertices to two disjoint classes such that each edge connects vertices which belong to distinct classes. Zschalig (2007) defines the Ferrers-graph of a formal context as follows:

Definition 3. *The* Ferrers-Graph *of a formal context* (G, M, I) *is* $\Gamma(I)$ *with*

set of vertices: $V(\Gamma(I)) = \bar{I}$ *with* $\bar{I} = G \times M \setminus I$ *and*

set of edges: $E(\Gamma(I)) = \{\{(a_1, b_2), (a_2, b_1)\} \mid (a_1, b_1), (a_2, b_2) \in I\}$.

The definition is easier to understand if one describes the formal context by a cross table as before (cf. Fig. 4 and Fig. 5). Then the empty cells of the table become the vertices of the Ferrers-graph, and two vertices are connected by an edge if and only if their cells violate the condition of a Ferrers-relation. Here, violating the condition of a Ferrers-relation means that the two 'partner' cells – which together with the two empty cells define the corners of a rectangle – both contain a cross. Hence, in the small example

×	
	×

the two empty cells are vertices of the corresponding Ferrers-graph, and they are connected by an edge:

×	•
•	×

Figure 11 demonstrates by the example of two edges how the Ferrers-graph of a formal context is constructed. In the left part of the figure, the two vertices $(2, c)$ and $(3, a)$ of the Ferrers-graph have to be connected by an edge since their partner cells $(2, a)$ and $(3, c)$ bear crosses. The right part of the figure demonstrates that the vertices $(3, b)$ and $(0, e)$ have to be connected by an edge too. As an example for two non-connected vertices of the Ferrers-graph consider the vertices $(2, c)$ and $(3, d)$. They are not connected by an edge in the Ferrers-graph since their partner cell $(2, d)$ does not bear a cross.

The whole Ferrers graph for this example context is given in Fig. 12. Here, the edges of the graph are labeled by the cells of the cross table of the formal context. Note that the Ferrers-graph is bipartite which is in accordance with the Hasse-planarity of the corresponding concept lattice shown in Fig. 12. However, the example set of classes is not S-sortable since the node labeled f does not lie on the S-graph of the concept lattice. Hence, by the main theorem on S-sortability it follows that the concept lattice of the enlarged set of classes is not Hasse-planar

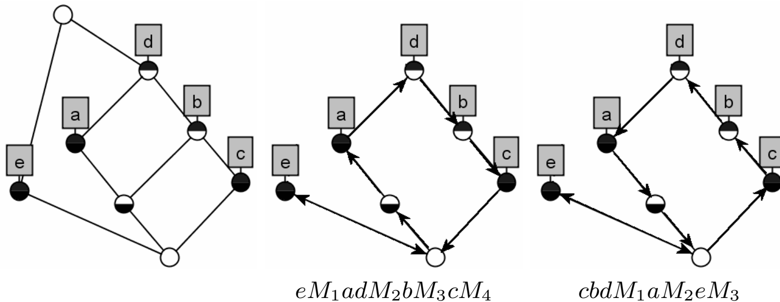
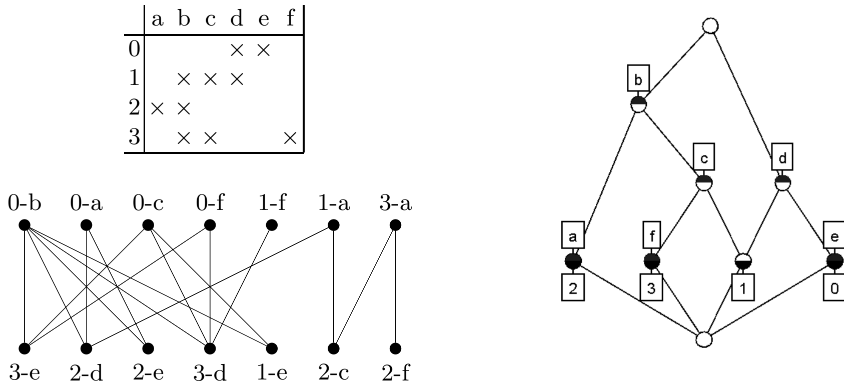


Fig. 10. Example with two distinct walks through the S-graph of which only the right one leads to an S-alphabet with minimal marker set (here, $\Phi = \{\{a, d\}, \{a, b, d\}, \{b, c, d\}, \{e\}\}$)

	a	b	c	d	e	f
0	•	•	•	×	×	•
1	•	×	×	×	•	•
2	×	×	•	•	•	•
3	•	×	×	•	•	×

	a	b	c	d	e	f
0	•	•	•	×	×	•
1	•	×	×	×	•	•
2	×	×	•	•	•	•
3	•	×	×	•	•	×

Fig. 11. Illustration for the definition of edges in Ferrers-graphs



	a	b	c	d	e	f
0				×	×	
1		×	×	×		
2	×	×				
3		×	×			×

Fig. 12. Example of a bipartite Ferrers-graph (upper left: formal context; lower left: Ferrers-graph; right: concept lattice)

and that its Ferrers-graph is not bipartite. Both, the enlarged concept lattice and its corresponding Ferrers-graph are given in Fig. 13. As demonstrated by the edge between the vertices 2-f and 9-b, the Ferrers-graph is not bipartite.

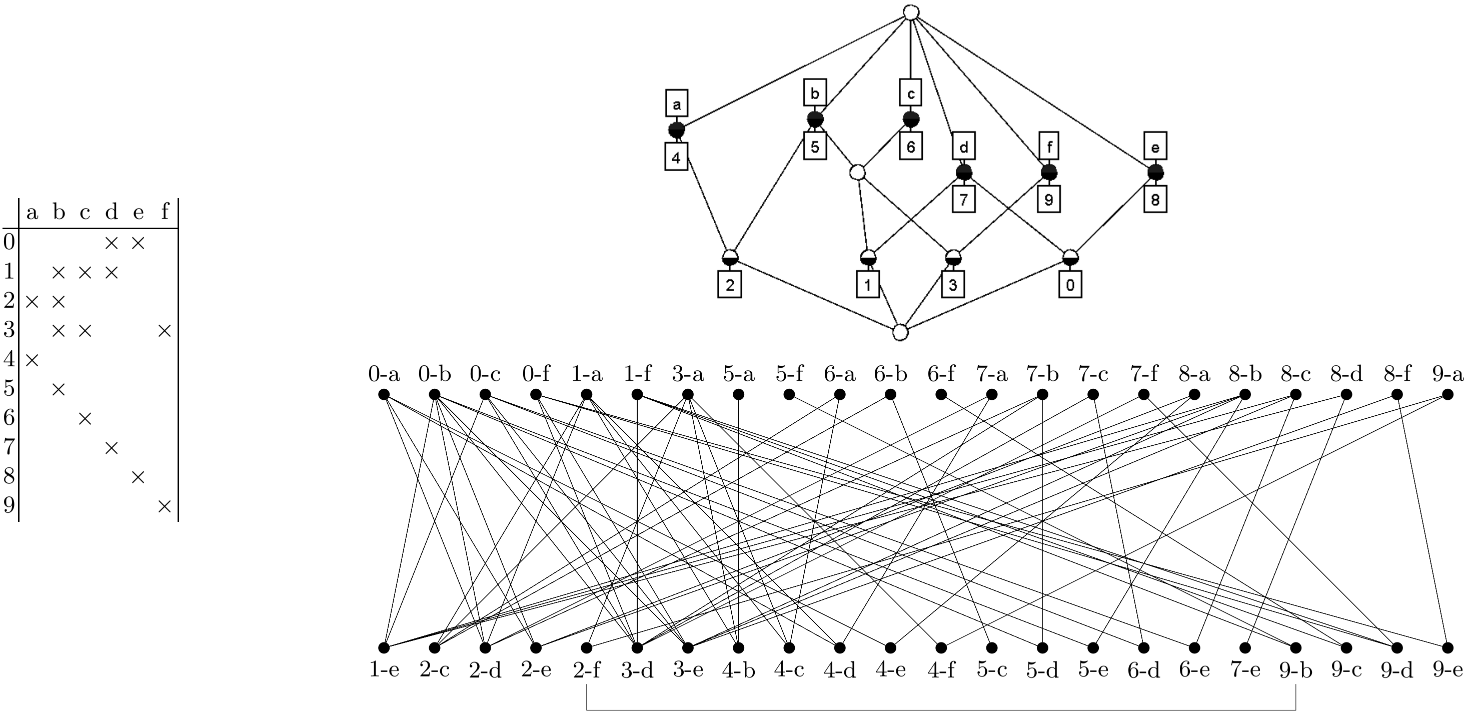
The Ferrers-graph of a formal context is bipartite if it is possible to assign its vertices to two disjoint classes such that each edge connects vertices which belong to different classes. This property can easily be algorithmically verified by assigning an arbitrary start vertex to the first class and assigning all vertices which are connected with it to the second class. Every time a vertex is assigned to one class, all neighbor vertices are assigned to the other class. This procedure has to be repeated for every connected component of the Ferrers-graph. If at any point a vertex has to be assigned to both classes, the Ferrers-graph is necessarily not bipartite. But if it is possible to assign all vertices to the classes without conflicts, the Ferrers-graph is bipartite. Hence, condition 3 offers a possibility to check algorithmically whether a set of classes is S-sortable or not.

3 Identifying Good Candidates for Duplication

The aim of this section is to illustrate how the three conditions for S-sortability can be applied in the construction of S-alphabets with duplications in the case of non S-sortable sets of classes. It turns out that for different sets of classes different strategies have to be chosen in order to tackle the problem of identifying those elements which have to be duplicated in order to get an S-alphabet with as few duplications as possible and a minimal number of markers.

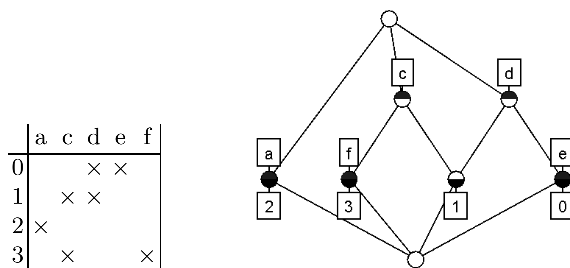
First, it will be demonstrated by the examples in Fig. 12 and Fig. 13 how in some cases condition 3 can be applied in order to identify minimal S-alphabets in the case of non-sortable sets of classes. Let therefore

$$\mathcal{A} = \{a, b, c, d, e, f\} \quad \text{and} \quad \Phi = \{\{d, e\}, \{b, c, d\}, \{a, b\}, \{b, c, f\}\} .$$



	a	b	c	d	e	f
0				×	×	
1		×	×	×		
2	×	×				
3		×	×			×
4	×					
5		×				
6			×			
7				×		
8					×	
9						×

Fig. 13. Example of a non-bipartite Ferrers-graph (left: formal context; upper right: concept lattice; lower right: Ferrers-graph



	a	c	d	e	f
0			×	×	
1		×	×		
2	×				
3		×			×

Fig. 14. Formal context and concept lattice of the set of classes in Fig. 12 reduced by b

As the Ferrers-graph in Fig. 13 of the enlarged set of classes is not bipartite, it is not possible to give an S-alphabet of $(\mathcal{A}, \Phi)$ without duplicated elements. The task is now to identify those elements whose duplication leads to an S-alphabet with as few duplicated elements as possible and a minimized marker set.

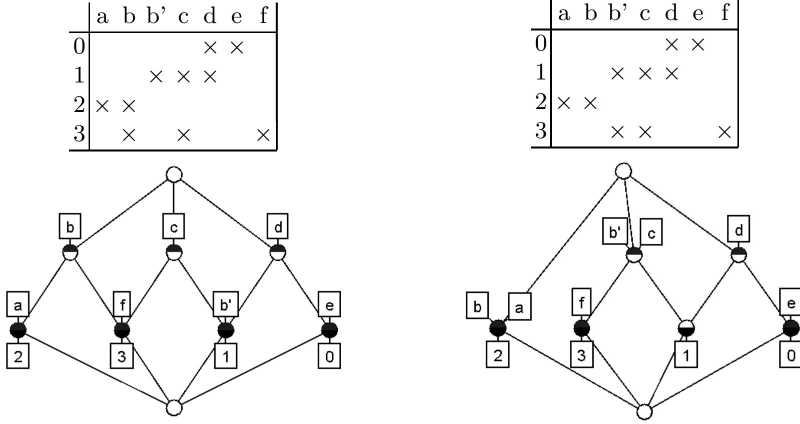
The Ferrers-graph in Fig. 13 indicates that the elements b and f cause the graph to be non-bipartite. Duplicating f would be pointless since f is an element of only one class, namely $\{b, c, f\}$. Therefore, it should be tried to duplicate b such that the set of classes gets S-sortable and thus S-encodable.

First, by condition 2 Fig. 14 indicates that the set of classes becomes S-sortable if b is completely removed. Furthermore, if a minimal S-alphabet can be gained by duplicating b, Fig. 14 restricts the order of the elements a, c, d, e and f. On the basis of the S-graph of the concept lattice in Fig. 14 the following four minimal S-alphabets of the set of classes reduced by b can be identified (the marker positions are indicated by vertical lines):

$$a|fc|d|e| \qquad fc|d|e|a| \qquad a|ed|c|f| \qquad ed|c|f|a| \ .$$

Adding a copy of b such that the resulting set of classes becomes S-sortable leads to one of the two formal contexts and corresponding concept lattices in Fig. 15. From both concept lattices one can read off S-alphabets with a minimum of four marker elements. As four markers are already needed in an S-alphabet of the set of classes reduced by b, all S-alphabets with minimal marker sets which can be read off the S-graphs of the two concept lattices in Fig. 15 are minimal (e.g., $ab|fc|b'd|e|$, $ed|b'c|fb|a$, $ab|fb'c|d|e|$, $ed|b'c|f|ab|$, ...).

The analysis of Ferrers-graphs is not always as informative as in the case of the discussed example, where one single edge can be identified that destroys the bipartity of the graph. Therefore, another example that involves a different method for the identification of good candidates for duplication will be presented: Let the set of classes consists of those sound classes used in Pāṇini's Sanskrit grammar that are denoted by pratyāhāras. The concept lattice of the corresponding pratyāhāra-context is given in Fig. 16. As mentioned before, it is not Hasse-planar, but proving that a graph like the one in Fig. 16 is not planar may be hard. In Petersen (2004) the proof is based on the *criterion of Kuratowski*, which states that a graph is planar, i.e. drawable without intersecting



	a	b	b’	c	d	e	f
0					×	×	
1			×	×	×		
2	×	×					
3		×		×			×

	a	b	b’	c	d	e	f
0					×	×	
1			×	×	×		
2	×	×					
3			×	×			×

Fig. 15. Formal contexts and concept lattices for possible duplications of b (cf. Fig. 12 and Fig. 13)

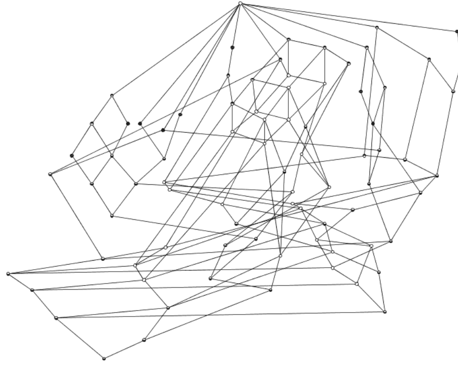
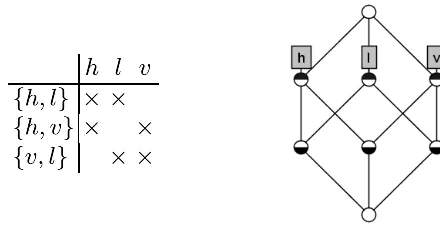


Fig. 16. Concept lattice of Pāṇini’s pratyāhāra-context

edges, if and only if it contains neither the graph nor the graph as a minor.[3] In Petersen (2004) it is shown that the graph in Fig. 16 has as a minor by identifying a fitting section of the graph.

Instead of applying Kuratowski’s criterion directly it is easier to work with a derived necessary condition for S-sortability: Figure 17 shows a context of three independent elements and its concept lattice. An ordered set like this concept lattice is Hasse-planar if and only if the graph which is the result of adding an extra edge connecting the bottom and the top node in its Hasse-diagram is planar. It can be easily verified that in the case of a concept lattice of three independent elements the resulting graph has as a minor. It follows that

[3] A graph is the *minor* of an other graph if it can be constructed from the latter by removing vertices and edges and contracting some of the remaining edges.



	h	l	v
$\{h,l\}$	×	×	
$\{h,v\}$	×		×
$\{v,l\}$		×	×

Fig. 17. Formal context of three independent elements and its concept lattice

whenever a set of classes has three independent elements it is not S-sortable. Three elements are said to be independent if for any pair of them there exists a set in the concept lattice which contains both, but not the third. The set of all independent triples of a formal context can be extracted algorithmically.

In the case of Pāṇini's pratyāhāra-context, one finds 249 independent triples. Interestingly, all of them include the sound h and no other sound element is included in all of them. Hence, h is the best candidate for duplication as h offers the only possibility to destroy all independent triples by a single duplication and thus to order the sounds in an S-alphabet with just one duplicated element. By an analysis of the concept lattice of the pratyāhāra-context reduced by h, it has been proven in Petersen (2008) that in the Śivasūtras Pāṇini has chosen a way of duplicating h that leads to a minimal S-alphabet.

4 Conclusion

The analyses of the various examples in this paper demonstrate how the three sufficient conditions for S-sortability offer different approaches for the construction of minimal S-alphabets which interlock and complement one another. As the problem of constructing minimal S-alphabets inherently bears the danger of combinatoric explosion, it is important to check which solution strategy is the most efficient in each individual case. One can benefit from the fact that the three conditions of S-sortability support different ways of tackling the problem. For instance, whether a graph is bipartite can be checked algorithmically while the question whether all labels lie on the S-graph of a concept lattice can be answered by simply looking at it. Hence, S-alphabets should be constructed semi-automatically by considering the application of all presented strategies.

In fact, deciding whether Pāṇini has actually chosen an optimal way of arranging the sounds in the Śivasūtras is more intricate than presented here. We have simplified the problem to the problem of constructing a minimal S-alphabet to the set of sound classes which are denoted by pratyāhāras in Pāṇini's grammar. But due to the following reasons this is not the exact problem which Pāṇini faced: First of all, not all sound classes in Pāṇini's grammar are denoted by pratyāhāras. For instance, Pāṇini also makes use of the older *varga*-classification of sounds, or sometimes he even simply lists the sounds involved in a rule. Second, Pāṇini permits overgeneralized rules by using a pratyāhāra in a rule that

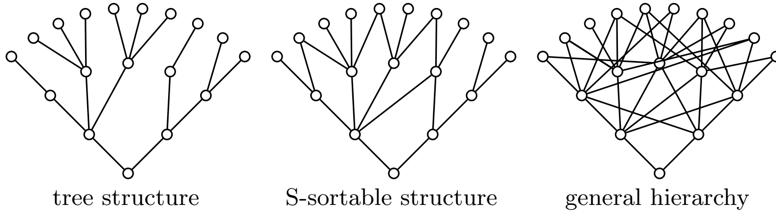


Fig. 18. Mid-position of S-sortable structures

denotes a larger class of sounds than the one to which the rule actually applies (cf. Kiparsky, 1991). Third, the order of the sounds in the Śivasūtras does not only depend on the classes which need to be encoded by pratyāhāras. A phonological rule, which claims that sounds of class A are replaced by sounds of class B, also has to ensure that a replaced element of class A is replaced by its counterpart of class B. In Pāṇini's grammar, a special meta-rule guarantees that the sounds are replaced by their counterparts according to their position in the sound lists denoted by the pratyāhāras (cf. footnote 1). Hence, a deeper analysis of the use of the sound classes in Pāṇini's grammar is still necessary in order to decide whether the Śivasūtras are optimal.

We will conclude by some remarks on the promising prospect of revitalizing Pāṇini's Śivasūtra-technique in order to approach some modern problems. A first field of problems is tackled in Petersen (submitted), namely the problem that quite often one is forced to order things linearly although they could be more naturally organized in a non-linear hierarchy (e.g., books on bookshelves, clothes on racks, . . .). S-orders may offer a way out as they order elements linearly, but in a sense bundle elements of one class up by keeping the distances between them small.

Another possible, but yet unexplored application area of the presented formalization of Pāṇini's technique is data representation in Computer Science. Data structures in Computer Science are encoded linearly as classical programming languages are inherently linear. Since tree structures can be encoded as nested lists, many formalisms only allow for tree structures and leave polyhierarchies out. However, in knowledge engineering multiple inheritance relations are central and thus polyhierarchies are badly needed. In this dilemma, S-sortable sets of classes could take over the position of tree structures due to the fact that they can be encoded linearly by lists with indexed brackets. Furthermore, they build up a hierarchical structure which takes a mid-position between tree structures and general hierarchical structures (cf. Fig. 18): They can be represented in a plane drawing like tree structures, but allow, at least in a limited way, for multiple inheritance like general polyhierarchies. A promising task is to explore to what extent Pāṇini's Śivasūtra-technique can be employed for the representation of hierarchies in order to allow at least for limited multiple inheritance without loosing the advantages of an efficient linear encoding and processing of hierarchical relations. The idea is to extend, for some tasks, the class of admissible

hierarchies from tree-shaped hierarchies to S-sortable ones. More on this idea can be found in Petersen (2008).

Acknowledgements. Part of the research presented in this paper was made possible by the Deutsche Forschungsgemeinschaft within the Forschergruppe FOR 600.

Bibliography

Böhtlingk, O.: Pāṇinis Grammatik. Leipzig (1887); Reprinted, Hildesheim (1964)

Cardona, G.: Studies in Indian Grammarians I: The Method of Description Reflected in the Śiva-Sūtras. Transactions of the American Philosophical Society 59(1), 3–48 (1969)

Faddegon, B.: The Mnemotechnics of Pāṇini's Grammar I: The Śiva-Sūtra. Acta Orientalia 7, 48–65 (1929)

Ganter, B., Wille, R.: Formal Concept Analysis. Mathematical Foundations, Berlin (1999)

Kiparsky, P.: Economy and the construction of the Śivasūtras. In: Deshpande, M.M., Bhate, S. (eds.) Pāṇinian Studies. Ann Arbor, Michigan (1991)

Kornai, A.: Mathematical Linguistics. Springer, London (2008)

Kornai, A.: The generative power of feature geometry. Annals of Mathematics and Artificial Intelligence (8), 37–46 (1993)

Misra, V.N.: The Descriptive Technique of Pāṇini. An Introduction. Mouton & Co., The Hague (1966)

Petersen, W.: A Mathematical Analysis of Pāṇini's Śivasūtras. Journal of Logic, Language, and Information 13(4), 471–489 (2004)

Petersen, W.: How formal concept lattices solve a problem of ancient linguistics. In: Dau, F., Mugnier, M.-L., Stumme, G. (eds.) ICCS 2005. LNCS, vol. 3596, pp. 337–352. Springer, Heidelberg (2005)

Petersen, W.: Linear coding of non-linear hierarchies – revitalization of an ancient classification method. In: Proceedings of the GfKl 2008 (submitted, 2008)

Petersen, W.: Zur Minimalität von Pāṇinis Śivasūtras – Eine Untersuchung mit Methoden der Formalen Begriffsanalyse. PhD thesis, University of Düsseldorf (2008)

Staal, F.J.: Artificial languages across sciences and civilizations. Journal of Indian Philosophy 34(1-2), 87–139 (2006)

Staal, F.J.: A method of linguistic description. Language 38, 1–10 (1962)

Staal, F.J.: The Sanskrit of science. Journal of Indian Philosophy 23(1), 73–127 (1995)

Zschalig, C.: Bipartite ferrers-graphs and planar concept lattices. In: Kuznetsov, S.O., Schmidt, S. (eds.) ICFCA 2007. LNCS, vol. 4390, pp. 313–327. Springer, Heidelberg (2007)

Tagging Classical Sanskrit Compounds

Brendan S. Gillon

McGill University
Montreal, Quebec

Abstract. The paper sets out a *prima facie* case for the claim that the classification of Sanskrit compounds in Pāṇinian tradition can be retrieved from a very slight augmentation of the usual enriched context free rules.

Keywords: *Aṣṭādhyāyī*, Classical Sanskrit, compounds, context free rules, Pāṇini.

1 Introduction

The aim of this paper is to make a *prima facie* case that the information pertaining to the grammar of compounds in Classical Sanskrit captured in their classification by the Pāṇinian tradition can be retrieved from a very slight augmentation of the usual enriched context free rules used by generative linguists. To make the complete case would require much more space than is available here.

I shall proceed as follows. First, I shall remind the reader of the general properties of compounds in Classical Sanskrit. Second, I shall set out what I mean by enriched context free rules. Third, I shall review the classification of compounds by the Pāṇinian tradition and show, for each category, how that category can be retrieved from the structure assigned to a compound by the enriched context free rules.

2 General Properties

The following are generally acknowledged regularities which the compounds of Classical Sanskrit exhibit.

1. Compounds are subject to the inflectional and derivational morphological forms of simple words (A 2.4.71; A 6.3.1; MBh to A 2.1.1, i.e., Kielhorn (ed) [5] v. I,p. 362.5; Whitney [8] §1246; and Cardona [1] pp. 264-265). In particular, inflection occurs at the end of compounds, not within them; derivational suffixes can be added as easily to compounds as they can be to words.
2. The accentuation of a compound is that of a simple word, not that of a phrase (A 6.1.158; MBh to A 2.1.1, i.e., Kielhorn (ed) [5] v. I, pp. 362.8-9; Whitney [8] §1246).

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 98–105, 2009.

3. Constituents of a compound, unlike constituents of phrases, have a fixed linear order (A 2.2.30; MBh to A 2.1.1, i.e., Kielhorn (ed) [5] v. I, p. 362.8; Cardona [1] pp. 261-264). In general, whereas no two immediate constituents of a compound can be transposed and the sense of the compound retained for its members; any two immediate constituents of a phrase can be transposed and the sense of the phrase retained.
4. Inflected words, which are external to a compound, are not construed with uninflected constituents subordinate within it (MBh to A 2.1.1).
5. A compound is usually analyzable into two immediate constituents (A 2.1.4); and if there is a head, it is the second immediate constituent (A 1.2.43; A 2.2.30; ; Whitney [8] §1246; Cardona [1] pp. 261-263).
6. Compounds are of unbounded complexity (Whitney [8] §1248).
7. A compound has a typical, and for Pāṇini, a canonical, phrasal paraphrase (*vigraha-vākya*) such that, if a compound has the form $[\mathrm{C\ D}]_i$ then its phrasal paraphrase has the form C_j D_i (where i and j denote one of the seven Sanskrit cases). Moreover, the head of a canonical phrasal paraphrase is the head of the compound being paraphrased.

The first four regularities make it plausible that lexical and phrasal syntax are distinct. The fifth and sixth regularities show that compounds in Classical Sanskrit are binary branching and recursive. Such are the kinds of structures which one would expect the enriched context free rules of the sort given by Selkirk (1982), among others, for English would generate.

To be sure, there are exceptions to these regularities, but happily they are not productive. For example, some compounds, such as conjunctive compounds (*dvandva* compounds) formed from the names of gods, do not have the accent of simple words. Moreover, there are cases where an inflectional affix occurs on a subordinate constituent within a compound (A 6.3.7-8), so-called *aluk* compounds, or within lexical derivation (A 6.3.17). However, these cases are not considered productive by any Sanskritist and they are best treated as items to be listed in the lexicon. (For examples and discussion, see Whitney [8] §1250 and Cardona [1], pp. 264-265.)

3 Enriched Context Free Rules

It is useful to distinguish between descriptive grammars and generative grammars. Descriptive grammars are those which state regularities which do not aim to generate all and only the well-formed expressions of the language. Traditional grammars of European languages and teaching grammars of various languages around the world are examples of such grammars. Generative grammars are those grammars which aim, on the basis of a lexicon and a set of rules, to generate all and only the acceptable expressions of a language. Pāṇini's *Aṣṭādhyāyī* is such a grammar, for it aims to do precisely that for Classical Sanskrit. Grammars of the American structuralists, what they called *constituency grammars*, are also examples of such grammars.

These grammars, though generative, were informal grammars. The first attempt to define and characterize formal grammars was undertaken by Chomsky [3][1], who distinguished regular grammars, from context free grammars, from context sensitive grammars, from unrestricted grammars. Chomsky [2] claimed that the constituency grammars of the American structuralists are properly formalized as context free grammars. However, as most formally minded linguists now recognize, this is not true. To be sure, American structuralist linguists did use rules which are instantiations of context free rules, but the rules they used had richer content, which included the use of features and of structured category labels.

Context free grammars can be viewed from a variety of perspectives. Initially, they were regarded as rewrite rules. The correlated labelled trees, then, represented equivalence classes of derivations. Eventually, linguists abandoned the view of context free grammars as sets of rewrite rules and adopted the view instead they characterized the order which the morphemes in a complex expression bear to one another. This is the view I shall take in what follows, though I suspect that nothing crucial depends on which of the two views one opts for. The latter view is simply the one which most linguists find more congenial.

It is now generally recognized that the labels for categories of expressions, features and subcategorization frames are all required in a grammar of a language. Enriched context free rules, then, are context free rules enriched with such additional structure. Let me state, then, some of the enrichments required for the treatment of expressions of Classical Sanskrit.[2] To begin with, one requires feature specification for case, number and gender. We shall not introduce these here, as they do not bear directly on the range of data to be addressed in this paper. In addition, one requires structured category labels. Basically, one requires labels for adjective, adverb, noun, preposition, verb and clause. However, these must be enriched so as to distinguish between stems, words (inflected stems) and phrases. The phrasal labels include A^1 (inflected adjective), A^2 (adjective phrase), N^1 (inflected noun), N^2 (noun phrase), V^1 (inflected verb), V^2 (verb phrase), P^1 (preposition), P^2 (prepositional phrase), D^1 (adverb), D^2 (adverbial phrase) and S (clause). The remaining labels, which are labels for stems, are A^0 (adjective stem), N^0 (noun stem), P^0 (preposition stem)[3] and V^0 (verb stem). (Since phrasal syntax plays only an incidental role here, I shall drop the superscripts and, unless otherwise specified X is short of X^0.) All compound stems will then have a label X (X^0). Subcategorization information is used to capture what morphologists call *bound morphemes*. I shall indicate bound morphemes with the linguist's customary use of a hyphen, preceding the morpheme, when the morpheme is suffixed to another constituent, and succeeding the morpheme,

[1] This publication was based on work done before 1957.

[2] It is such enrichments which are at the heart of such grammars as Generalized Phrase Structure Grammar (GPSG) and Head Driven Phrase Structure Grammar (HPSG).

[3] There is probably no need to distinguish between preposition stem and prepositional word.

when it is prefixed. (How subcategorization is to be handled is a much more complex matter, involving many facets of the grammar not discussed here.)

4 Traditional Classification

Pāṇini's treatment of compounds in his *Aṣṭādhyāyī* is one familiar to contemporary linguists. Each compound is paired with a canonical phrasal paraphrase (*vigraha-vākya*). Underlying the compound and its canonical phrasal paraphrase is a string of symbols, which denote what the compound and its paraphrase denote. At a suitable point in the derivation, elements corresponding to the inflection of words in the paraphrase are optionally deleted. As a result of this correspondence between compound and canonical paraphrase, it is possible to classify the compounds using properties of the paraphrases. This, then, is the basis for the classification of compounds used in the later Pāṇinian tradition. Here is the classification:

1. *aluk*
2. *luk*
   (a) *avyayībhāva*
   (b) *dvandva*
   (c) *tatpuruṣa*
      i. *nañ tatpuruṣa*
      ii. *prādi tatpuruṣa*
      iii. *upapada tatpuruṣa*
      iv. *vibhakti tatpuruṣa*
      v. *karmadhāraya*
         A. *viśeṣaṇa-pūrva-pada-karmadhāraya*
         B. *viśeṣaṇa-uttara-pada-karmadhāraya*
         C. *viśeṣaṇa-ubhaya-pada-karmadhāraya*
         D. *upamāna-pūrva-pada-karmadhāraya*
         E. *avadhāraṇa-pūrva-pada-karmadhāraya*
         F. *upamāna-uttara-pada-karmadhāraya*
   (d) *bahuvrīhi compounds*

The question is: can one recover from the analysis of a compound by enriched context free rules the classification of the compound within this schema. The answer seems to be yes. However, the full case cannot be made here, both because the data to be surveyed demands a paper much longer than what is permitted and because the problem of compound analysis is intimately connected with other aspects of the grammar whose precise treatment remains either obscure or undecided. Nonetheless, I shall sketch out the basic ideas, filling in, as required, ancillary assumptions.

The simplest cases are the *aluk* compounds. These are the compounds in which the left sister is inflected. Consider, for example, *ātmanepada*. It is analyzed as [$_N$ [$_{N^1}$ *ātmane*] [$_N$ *pada*]].[4] The fact that the analysis contains a bracket labelled with N^1 as a left sister to a bracket labelled with N is both necessary and sufficient to identify the compound as an *aluk* compound.

All other compounds are *luk* compounds, meaning the left sister constituents are all stems, either bound or unbound. Within the *luk* compounds, *avyayībhāva* compounds are easily identified by their analysis. They are compound stems inflected as adverbs, as is, for example, [$_{D^1}$ [$_P$ *upari*] [$_N$ *bhūmi*]].

Also easily identifiable from their parses are *nañ-tatpuruṣa* compounds, *upapada-tatpuruṣa* compounds and *prādi-tatpuruṣa* compounds. *Nañ-tatpuruṣa* compounds are compounds prefixed with the bound morpheme *a*- or *an*-. Thus, for example, it is evident that [$_N$ [$_A$ *a*-] [$_N$ *brāhmaṇa*]] and [$_N$ [$_A$ *an*-] [$_N$ *aśva*]] are such compounds. An *upapada-tatpuruṣa* compound is one whose right sister is a bound, nominal morpheme derived from a verbal root. Examples of such bound morphemes are: -*bhid*, -*jña*, -*stha*, -*dṛś*, -*ghna*, -*cara*, etc.

On the assumption that one can identify a bound, nominal morpheme derived from the verbal root, one can easily identify a compound such as [$_N$ [$_N$ *sarva*] [$_N$ -*jña*]] as an *upapada-tatpuruṣa* compound. Finally, a *prādi-tatpuruṣa* compound is one whose first constituent is either a preposition (e.g., *pra*), a prefixing bound morpheme (e.g., *ku*-) or an indeclinable (e.g., *puras*). These morphemes are listed in the grammar with the first member of the list being the preposition *pra*. (Hence, they are given the name *prādi* compounds.) Each of these compounds, then, are readily identified from their analysis and the *prādi* list. Examples of such compounds are: [$_N$ [$_P$ *adhi*] [$_N$ *rāja*]], [$_N$ [$_A$ *ku*-] [$_N$ *puruṣa*]] and [$_N$ [$_A$ *puras*] [$_N$ -*kāra*]].

We now come to compounds which require further annotation for their identification. They are those compounds comprising two adjectival stems, two nominal stems or a nominal stem followed by an adjectival stem.[5] Let us begin with stems of the form N N. *Dvandva* compounds, many *vibhakti-tatpuruṣa* compounds and *karmadhāraya* compounds are of this form. It is common to distinguish headed constituents from non-headed constituents. *Vibhakti-tatpuruṣa* compounds and most *karmadhāraya* compounds are headed, indeed, right headed. *Dvandva* compounds are non-headed compounds; and some *karmadhāraya* compounds are also non-headed.

The simplest extension of the notation is to introduce a special symbol for the non-headed compounds, inserting between the elements a plus sign, say. Thus, one would have the *dvandva* compound [$_N$ [$_N$ *rāma*]+[$_N$ *kṛṣṇa*]]. Some so-called *viśeṣaṇa-ubhaya-pada-karmadhāraya* compounds are also non-headed: for example, [$_A$ [$_A$ *snāta*]+[$_A$ *anulipta*]]. Now it is easy from this notation to see

[4] Since *aluk* compounds are not productive, they are listed in the dictionary and they can be listed with their analysis. Some questions of implementation arise with respect to whether or not one wishes to encode the case of the inflected subordinate word.

[5] I am making the simplifying assumption that participial stems are labelled as adjectival stems.

which is which. The compounds which have the plus sign and both of whose constituents are nouns are *dvandva* compounds; other compounds with the plus sign are *viśeṣaṇa-ubhaya-pada-karmadhāraya* compounds.

To distinguish among the remaining compounds, one could do the following: one could introduce a labelled relational symbol.[6] This symbol could be indexed by a numeral between one and seven. Each *vibhakti tatpuruṣa* would have the numeral corresponding to its case. Thus, one has [$_N$ [$_N$ *sukha*] $\leq_2$ [$_A$ *āpanna*]], [$_N$ [$_N$ *ākhu*] $\leq_3$ [$_A$ *daṁśita*]], [$_N$ [$_N$ *go*] $\leq_4$ [$_A$ *hita*]], [$_N$ [$_N$ *vṛka*] $\leq_5$ [$_A$ *bhīta*]], [$_N$ [$_N$ *rāja*] $\leq_6$ [$_N$ *puruṣa*]] and [$_N$ [$_N$ *īśvara*] $\leq_7$ [$_A$ *adhīna*]].

Those with the numeral one are identifiable as *karmadhāraya* compounds. In this way, the *viśeṣaṇa-pūrva-pada-karmadhāraya* compounds, as exemplified by the compound [$_N$ [$_A$ *dīrgha*] $\leq_1$ [$_N$ *kaṇṭha*]], the *viśeṣaṇa-ubhaya-pada-karmadhāraya* compounds, as exemplied by [$_A$ [$_A$ *tulya*] $\leq_1$ [$_A$ *śveta*]], the *upamāna-pūrva-pada-karmadhāraya*, as exemplified by [$_N$ [$_N$ *anala*] $\leq_1$ [$_N$ *uṣṇa*]], the *avadhāraṇa-pūrva-pada-karmadhāraya*, as exemplified by [$_N$ [$_N$ *rāja*] $\leq_1$ [$_N$ *ṛṣi*]], and the *upamāna-uttara-pada-karmadhāraya*, as exemplified by [$_N$ [$_N$ *puruṣa*] $\leq_1$ [$_N$ *vyāghra*]] would all be identifiable as *karmadhāraya* compounds. I shall leave open the question as to how these subclasses might be distinguished from one another using the enriched context free notation advocated here.

Finally, we come to *bahuvrīhi* compounds. Such compounds are most easily identified by the fact that their final constituent is a noun but they behave like adjectives. When this is marked inflectionally, such compounds are easily identified. But this is not always so. For example, when a *bahuvrīhi* compound is the left sister of a compound or of a derivational suffix, it cannot be grammatically identified. Indeed, such cases are ambiguous; and one must rely on the context to figure out whether the compound is a *tatpuruṣa* or a *bahuvrīhi*. Moreover, if the last word of the *bahuvrīhi* compound is of the same gender as the noun it modifies, again it is ambiguous. The easiest way to annotate such compounds is with a phonetically null suffix (see Gillon [4] for discussion), but how precisely to implement that depends on how inflectional tagging is to be done, another complexity not addressed here.

5 Bound Morphemes

We have already seen some instances of the utility of the notion of a bound morpheme, to be formalized as subcategorization, in providing parses for Sanskrit compounds. I end the paper with an indication of still further uses. To begin with, consider the stems *pūrva*, *apara*, *adhara*, *uttara*, *ardha* and *madhya*. In compounds, they are adjectives, in phrases they are nouns. This distinction is nicely handled by treating the adjectives as bound morphemes and the nouns as unbound morphemes.

In addition, many words, which, when uncompounded, belong to one inflectional class, belong to another, typically the *a* stem inflectional class, when

---

[6] In earlier work, I used the symbol $\prec$.

compounded. In each case, subcategorization can be used to handle the treatment of these stems.

1. The word *ṛc*, which has a consonantal stem, becomes the *a*-stem *ṛca*, when preceded by a word in a compound. The same holds for the words *pur* (*pura*), *ap* (*apa*), *dhur* (*dhura*) and *pathin* (*patha*).
2. The consonantal stem words *sāman* and *loman*, when preceded by the prepositions *prati*, *anu* and *ava* become the *a*-stem words *sāma* and *loma*, respectively.
3. The *i* stem word *bhūmi* becomes the *a* stem word *bhūma*, when preceded in compound by either *kṛṣṇa*, *pāṇḍu* or a numeral.
4. The *ī* stem words *nadī* and *godavarī* become the *a* stem words *nada* and *godavara*, respectively, when preceded in compound by a numeral.
5. The *r* stem word *catur* becomes the *a* stem word *catura*, when preceded in compound by either *tri* or *upa*.
6. The *s* stem word *varcas* becomes *a* stem word *varcasa*, when preceded in compound by either *brahman*, *hastin*, *palya* or *rājan*.
7. The *s* stem word *tamas* becomes *a* stem word *tamasa*, when preceded in compound by *ava*, *sam* or *andha*.
8. The *n* stem word *adhvan* becomes *a* stem word *adhva*, when preceded in compound by a preposition.
9. The *i* stem word *aṅguli* becomes *a* stem word *aṅgula*, when preceded in compound by either a numeral or an indeclinable.
10. The *i* stem word *rātri* becomes *a* stem word *rātra*, when preceded in compound by either a numeral or an indeclinable or *ahan* or *sarva* or a word denoting part of the night.
11. The *n* stem word *ahan* becomes *a* stem word *aha*, when preceded in compound by either a numeral or an indeclinable or *sarva* or a word denoting part of the day.

6 Conclusion

Above, I have made a *prima facie* case that the information pertaining to the grammar of compounds in Classical Sanskrit captured in their classification by the Pāṇinian tradition can be retrieved from a very slight augmentation of the usual enriched context free rules used by generative linguists. I have reviewed this classification and I have shown, for each category in the classification, how that classification can be retrieved from a fairly standard set of enriched context free rules, adapted for Classical Sanskrit.

References

[1] Cardona, G.: Pāṇini: his work and its traditions. Background and Introduction. Motilal Banarsidass, New Delhi (1988)
[2] Chomsky, N.: Syntactic structures. Mouton and Company, The Hague (1957) (Janua Linguarum: Series Minor n. 4)

[3] Chomsky, N.: Formal properties of grammars. In: Luce, R.D., Bush, R., Galanter, E. (eds.), vol. 2, pp. 323–418 (1963)

[4] Gillon, B.S.: Autonomy of word formation: evidence from Classical Sanskrit. Indian Linguistics 56(1–4), 15–52 (1995)

[5] Kielhorn, F. (ed.): The Vyākaraṇa Mahābhāṣya of Patañjali, 4th edn. Bhandarkar Oriental Research Institute, India (1880) (revised by R. N. Dandekar, 1985)

[6] Luce, R.D., Bush, R., Galanter, E.(eds.): Handbook of mathematical psychology. Wiley, New York (1963)

[7] Selkirk, E.O.: The syntax of words. The MIT Press, Cambridge (1982)

[8] Whitney, W.D.: Sanskrit grammar: including both the classical language, and the older dialects, of Veda and Brahmana, 2nd edn. Harvard University Press, Cambridge (1881) (11th reprint, 1967)

Extracting Dependency Trees from Sanskrit Texts

Oliver Hellwig

Institut für Sprachen und Kulturen Südasiens, Freie Universität Berlin, Germany

Abstract. In this paper, I describe a hybrid dependency tree parser for Sanskrit sentences improving on a purely lexical parsing approach through simple syntactic rules and grammatical information. The performance of the parser is demonstrated on a group of sentences from epic literature.

Keywords: Sanskrit syntax, Sanskrit word order, dependency tree.

1 Introduction

Designing a reliable algorithm for the automatic syntactic analysis of Sanskrit phrases is an important, though still unsolved problem in computational linguistics. Some previous approaches to this problem are based on built-in rules that encode the syntax of regular Sanskrit phrases using, for instance, finite automata. How to find these syntactic rules is not often the central focus of interest, however. Many researchers use *kāraka* analysis (e.g., [6], claiming free word order for Sanskrit) or the syntax rules formulated in modern learner's manuals such as Apte's or Kale's grammars. Whether these rules describe the correct word order of classical Sanskrit texts remains open to discussion, however, because they may reflect either a pre- (Pāṇini's *kāraka* theory) or post-classical use of Sanskrit. Modern Indological research is no great help in finding the syntactic rules of the classical language. German Indologists such as Delbrück [4], Speyer [9], Canedo [2] and, more recently, Ickler [8] took great pains in analyzing large corpora in detail, but they concentrated on Vedic and pre-classical prose. Results of these stylistic studies of early Sanskrit are hardly applicable to classical texts, which are, in addition, written in verse in most cases (cmp. the critical remarks in [5] on the bias in selecting the texts). The same holds for Staal's frequently cited work [10], which does not care very much about the word order in real Sanskrit texts. In summary, the syntactic rules used in rule-based approaches are derived from theories about modern or pre-classical Sanskrit and then applied to the classical language. This procedure implies that Sanskrit syntax has remained unchanged over an interval of over 3000 years, a claim well suiting the tendency to deny any development in this language, but not founded on large-scale research.

At this point, we find ourselves in a chicken-and-egg dilemma. Before using a rule-based approach to analyze the syntax of classical texts, we need the syntactic

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 106–115, 2009.

rules for the classical language. To find these rules in significant numbers, we need a syntactic parser that is, as just described, rule-based in many cases. We may, therefore, simplify the starting problem: The aim is to design not a complete syntactic parser, but instead an algorithm giving the most probable dependency tree of a lexically and morphologically analyzed sentence. Valid rules describing the word order of classical Sanskrit may be derived from a large number of manually corrected trees in a later step; however, this step is not discussed in this paper.

Among the numerous approaches to derive dependency trees from sentences in natural languages, a recently published thesis by Yuret is especially interesting because it combines an appealing basic idea with an unsupervised learning algorithm [12]. In this paper, I describe how the performance of Yuret's purely lexical parser can be improved through the addition of some simple, yet efficient, features such as information about grammar (2.2) and valences (2.2), smoothing probabilities (2.2) and a few fixed syntactic rules (2.2). Because test data for Sanskrit syntax are not available, the performance of the parser is demonstrated during the improvement steps by analyzing a few sample sentences from the Rāmāyaṇa (2.3).

2 Building a Lexicalized Parser

2.1 Yuret's Model

The starting point of the following experiments is the purely lexical dependency parser described by Yuret [12]. According to Yuret, syntactic information is captured by the mutual information between the lexical components of a sentence (cmp. [12, 26-31] and Footnote 1, below). Because mutual information does not depend on the order of two lexical items, the dependency structure constructed using Yuret's algorithm is undirected (but may, of course, be transformed into a directed graph as soon as the root item is identified). Furthermore, the dependency structure is acyclic and, therefore, a tree, which is equivalent to the claim that each word in a sentence should have only one governing word. Finally, Yuret claims that the dependency structure should be projective, i.e., its links should not cross. The author admits that this restriction holds only for many (not all!) sentences in natural languages; this warning is especially important for Sanskrit verses. However, projectivity simplifies the construction of the tree to such a degree that the occasional errors caused by this condition may be neglected. In summary, the application of Yuret's ideas to a sentence produces a dependency tree that reflects the syntactic structure of the sentence as given by its lexical components. The arrangement of such a tree is not necessarily identical to the structure found using, for instance, constituent analysis. However, items close to each other in the dependency tree should constitute syntactic units in regular constituent analysis.

Yuret's system consists of two parts. The *processor* builds dependency structures for a series of lexical units (= a sentence), while the *learner* represents

the "knowlegde" or "brain" storing information gained from previously analyzed sentences. To parse a sentence, the processor searches for the most probable structure given the information stored in the learner. Although there is a Viterbi style algorithm calculating the exact solution, Yuret proposes the following approximation that reduces computation time from $O(n^5)$ to $O(n^2)$ [12, 35ff.]:

1. Each sentence is read from left to right. For each word w_i to the left of the current word w_j, the conditional probability $p(w_j|w_i) = \frac{p(w_i,w_j)}{p(w_i)}$ is calculated.[1]
2. If $p(w_j|w_i) \neq 0$, the words w_i and w_j may be connected with a link. However, this link can only be created if (1) it does not intersect with an already existing link and (2) it does not create a cycle in the dependency structure. If condition (1) or (2) applies, the new link is only inserted if its value $p(w_j|w_i)$ is higher than the respective value of the existing link. In this case, the existing link is removed from the dependency tree.
3. When the sentence has been analyzed, the learner is updated with the linking information created in the second step. Therefore, only the cooccurrence frequencies of words being connected by a link are increased in the learner. During the first cycles of the algorithm, the "brain" of the learner is empty and no dependency structure can be created. In these cases, the learner is updated using pairs of adjacent words.

2.2 Improving Yuret's Parser

After being trained on the current corpus of the `SanskritTagger` database (cmp. [7]), Yuret's parser is able to identify some syntactic substructures in unknown

[1] Yuret uses the mutual information of ordered pairs of words $MI(w_i, w_j) = \log \frac{p(w_i,w_j)}{p(w_i,*)\cdot p(*,w_j)}$ instead of the conditional probability. If L denotes the dependency structure of a sentence S, W the set of all words w_i contained in S, and w_0 the head word of the structure, Yuret explains the use of mutual information as follows [12, 28/29]. The joint probability of the entire sentence is given by

$$p(S) = p(L)p(w_0) \prod_{(w_i,w_j)\in L} p(w_j|w_i) = p(L)p(w_0) \prod_{(w_i,w_j)\in L} \frac{p(w_i, w_j)}{p(w_i)}.$$

Because a projective tree constructed from a sentence S has $|S|-1$ connections after the head word has been identified, this expression can be rewritten as

$$p(S) = p(L) \prod_{w_i\in W} p(w_i) \prod_{(w_i,w_j)\in L} \frac{p(w_i, w_j)}{p(w_i,*)\cdot p(*,w_j)},$$

demonstrating, according to Yuret, that the syntactic information of a sentence can be expressed by the mutual information contained in the lexemes that constitute the sentence. Following Yuret, I set $p(S)$ to a constant factor. – For reasons of numerical accuracy, I use the logarithm of $p(w_j|w_i)$ instead of the raw value (multiplication $\rightarrow$ addition).

sentences. Nevertheless, when we analyze the sample phrases (cmp. 2.3), it becomes obvious that the parser never manages to identify the correct overall structure of any of the samples. This is probably caused by the comparatively small number of data used for training the parser. While Yuret reports convincing results only for databases of more than 10 million words, the `SanskritTagger` database comprises about 2.5 million lexical units. This number is probably not large enough to arrive at reliable estimations of lexical cooccurrence. The following sections describe how to improve the performance of the parser without training it on more lexical data.

Grammatical information. Every word stored in the `SanskritTagger` database is accompanied by grammatical information concerning, for instance, number, case or tense. Although Yuret deliberately excluded this kind of information from the parsing process, I have observed clearly superior analysis of the sample phrases when grammatical information was taken into account. To include grammar, the conditional probability of pairs of lexical items is multiplied by the conditional probability of the respective grammatical categories. These categories are a simplified version of those described in [7, 44] since only person and number are recorded for verbal forms.

Verbal valences. The second improvement on Yuret's model concerns finite verbal forms and their preferred valences. In many cases, verbs show a strong preference for certain cases, which may be used to enforce links between verbs and their valences. For this sake, we have built from the training data a verb-valence dictionary that stores verbs and the cases that typically occur close to them. Before starting the main learning process, the part of the corpus used for training is scanned for verbal forms. If a finite verb is encountered we search for the next and previous two declined nouns that are included in the same sentence as the verb, and store the verb-case combination in a preliminary table. Next, we calculate the global relative frequencies of all cases and then find extraordinary strong verb-case combinations. A combination is considered strong if (1) it occurs with a frequency of more than 15% and the verb is referenced at least 100 times, or (2) the relative frequency of case c for verb i is significantly larger than the global relative frequency for this case. If a_c is the global average frequency of case c, f_{ci} is the frequency of case c given verb i, and n_i is the sum of all f_{ci} for verb i, we use a simple χ^2 test, with significance assigned at the 10% level ($\chi^2 \geq 1.64$, 1 df), to assess whether f_{ci} is significantly above the expected number of occurrences of case c:

$$\chi^2 = \frac{(f_{ci} - a_c \cdot n_i)^2}{a_c \cdot n_i} + \frac{((n_i - f_{ci}) - (1 - a_c) \cdot n_i)^2}{(1 - a_c) \cdot n_i}$$

If one of these two conditions is fulfilled, the verb and case are stored in a separate valence dictionary.

Whenever the combination of a verb and a declined noun is found during the processing of a new sentence, we search for the verb in the valence dictionary just described. If the case of the declined noun is among the favorite cases of the verb,

the linking strength between the verb and noun is enforced by a positive value (see page 111 for details). – Given the importance of valence information for parsing (see, for instance, [11]), the valence dictionary could be corrected manually in a future version of the program and even enriched by lexical information concerning preferred valences.

Smoothing cooccurrence frequencies. One of the main problems in processing natural language using statistical methods is the sparseness of data, especially of n-grams of higher order. Among the many proposed solutions to this problem, we find simple strategies such as linear interpolation (see, e.g., [1]) or add-alpha smoothing. A more sophisticated strategy that makes use of lexical information gained from the training corpus was proposed by Dagan et al. [3], and it was successfully applied to the parsing of Sanskrit sentences. This strategy consists of two steps: a preprocessing step, during which similar words are retrieved from training data, and the actual smoothing step. During preprocessing, we calculate the Kullback-Leibler divergence D between all pairs of words w_i and w_j that are contained in the corpus C:

$$D_{ij} = \sum_{w_k \in C} p(w_k|w_i) \log \frac{p(w_k|w_i)}{p(w_k|w_j)}.$$

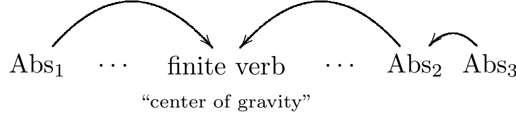
Based on D_{ij}, the n nearest words w_j are stored for each w_i. If an unknown bigram (w_1, w_2) is found during parsing, the conditional probability $p(w_2|w_1)$ is estimated using conditional probabilities of word pairs (w'_1, w_2), where w'_1 is a word similar to w_1 found in the preprocessing step. If S_u is the set consisting of the n words w_u nearest to w_1, the estimation $p^*(w_2|w_1)$ is calculated in the following way:

$$p^*(w_2|w_1) = \sum_{w_u \in S_u} p(w_2|w_u) \frac{D_{1u}}{\sum_{w_u \in S_u} D_{1u}}$$

Using simple syntactic rules. The final and, in my opinion, most effective way to improve Yuret's model is the use of simple syntactic rules, which transform the lexicalized base model into a hybrid parsing approach. I distinguish between two types of syntactic rules: *Fixed* rules encode the syntactic structures of a phrase that are certain to occur (given that the phrase is complete). These rules create fixed links or prevent lexical links from being constructed. In addition to describing the basic syntactic structures of a sentence, these rules strongly reduce the number of possible links and thereby suppress improbable analyses. On the other hand, *enforcements* increase or decrease the linking strength, but they do not insert or remove links from the dependency tree.

Fixed rules can further be divided into positive and negative rules. *Positive* rules describe the basic structure of a sentence containing exactly one verb, which may be supplemented by a congruent subject and absolutives. To detect this basic structure, the sentence is repeatedly scanned from left to right. In the first scan, the finite verb is detected and linked to the beginning of the phrase

(→ head-verb). Next, absolutives are found and connected either to the head-verb or to other absolutives. The head verb serves as a "center of gravity" indicating the search direction from each absolutive contained in the sentence:



Finally, nominatives congruent with the head verb are connected to it. If several nouns can be connected with the verb, the most probable one given the lexical attraction between noun and verb is selected. *Negative* or restrictive rules prevent possible links from being inserted into the dependency tree. Currently, three negative rules are used:

1. Words contained in a composite must only be linked to the head of the composite or other words contained in the composite. – This rule describes the correct formation of composites, but it is sometimes neglected in real texts. To give just one example: Someone "whose body is pierced by arrows" should be a *śaraviddhaśarīraḥ*. However, expressions such as *śarair viddhaśarīraḥ* can be frequently found, for instance, in epic texts.
2. An indeclinable must not be linked with a noun or adjective. – Exceptions are indeclinables forming part of composites such as *su-* or *nānā-*.
3. Incongruent nouns, except for the combination *any case-genitive*, must not be linked. – The validity of this rule is not established for the nominal style of scientific Sanskrit; see, e.g., *prasiddhasādharmyāt sādhyasādhanam upamānam* (NyāSū, 1, 1, 6), where *-sādharmyāt* is dependent on *-sādhanam*.

Enforcements change the strength of a new link, but they do not influence its insertion directly. We have met the first type of enforcement in section 2.2: If a case is among the preferred valences of a finite verb or absolutive, the link between the noun and the verb is enforced. In addition, the following three syntactic enforcements are used:

1. The linking strength between grammatically congruent nouns is increased (e.g., *tena* $\overset{+}{\leftrightarrow}$ *balena*, *akṣayasya* $\overset{+}{\leftrightarrow}$ *ātmanaḥ*).
2. Links between nominatives not identified as subject (see above) and congruent finite verbs are enforced.
3. In the early phases of learning, indeclinables have a strong influence on the lexical information due to their high frequencies. Therefore, the strength of a link connecting an indeclinable with any other word (except for finite verbal forms) is weakened. This enforcement is only applied when negative rule 2 is not valid.

The parameter values for the four enforcements (including the combination of verb and valence from section 2.2) were estimated using a genetic algorithm. Running this algorithm repeatedly for 100 generations resulted in the following average parameter values: verb - valence: 3.2, congruent nouns or adjectives, nominative - verb: 3, indeclinables: 0.7.

2.3 Evaluation

In this paper, we cannot present a true evaluation because test data for Sanskrit syntax are not available. Therefore, we demonstrate the performance of the parser using two sentences from chapter Rāmāyaṇa, Bālakāṇḍa 9, which was excluded from the training set, and the popular benchmark sentence *pramāṇabhūta ācāryo darbhapavitrapāṇiḥ prāṅmukhaḥ śucau avakāśe upaviśya mahatā yatnena sūtraṃ praṇayati sma*. The two sentences from Rāmāyaṇa are:
Rām, Bā, 9, 6: *śrutvā tatheti rājā ca pratyuvāca purohitam.*
and
Rām, Bā, 9, 32: *evaṃ sa nyavasat tatra sarvakāmaiḥ supūjitaḥ.*

Figure 1 shows the results of parsing the two sample phrases from the Rāmāyaṇa using Yuret's basic model. As mentioned on page 108, the parser is not able to identify even the basic structures of the sentences, possibly due to the small number of training data. In addition, the strong influence of indeclinables on the dependency structure is clearly discernible. In Figure 2, the same two sentences are parsed with grammatical information (2.2), valences (2.2), smoothing (2.2) and syntactic *enforcements* (2.2) activated. Although the parser is still far from able to identify the correct structure of the sentences, it found some important substructures such as *pratyuvāca* ↔ *purohitam*, *tathā* ↔ *iti* (*iti* terminating a direct speech) and the complex made of two composites in the second sentence. As becomes apparent from intermediary stages of learning not reproduced in this paper, the detection of the last substructure was especially influenced by the valence dictionary. In the last test, whose results are displayed in Figure 3, all optimizations are activated. Now, each of the sentences contains only one error. The word *ca* should probably not be connected to the head verb in the first sentence. In the second sentence, *tatra* remains unconnected to the rest of the sentence (but could, of course, easily be associated with *nyavasat* after finishing the parsing process).

Parsing the "benchmark sentence" *pramāṇabhūtaḥ . . .* results in equally good analysis (cmp. Figure 4). After fixing *praṇayati* as the head verb, the algorithm connects the absolutive *upaviśya* to the verb and selects the composite ending in *-pāṇiḥ* as the subject of the sentence. Here, a human user would certainly select *ācāryaḥ*, and, in some runs of the learning process, this word is indeed marked as the subject of the sentence. These differences can be explained by the heuristic nature of the learning process and can perhaps be amended by running the process repeatedly with different initalizations and then averaging the results. Among the remaining substructures, attention should be paid to the adverbial expressions modifying the absolutive and the head verb. Both adverbial structures are connected to the right verb and are, in addition, sorted correctly (*yatnena* modifies *praṇayati* and is itself modified by *mahatā*, etc.). How the nominatives in the beginning of the sentence are connected remains open to discussion even for a human user. However, it should be noted that the parser correctly associates the directional adjective *prāṅmukhaḥ* with the absolutive *upaviśya* and not with the preceeding and congruent nominative *-pāṇiḥ*. On the whole, the few restrictions introduced by the fixed syntactic rules clearly improve the analysis of the sentences.

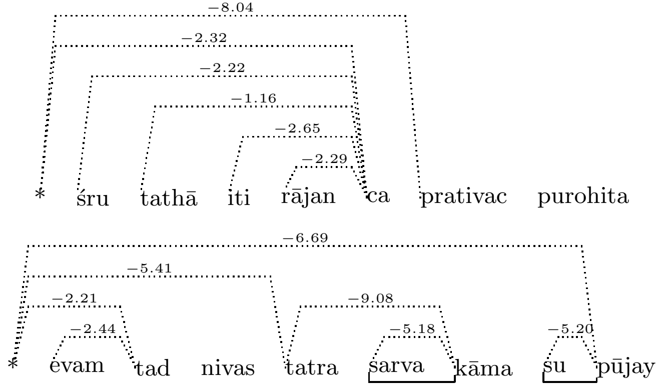


Fig. 1. Sample phrases parsed using Yuret's method – The numbers give the logarithm of the conditional lexical probability of two words

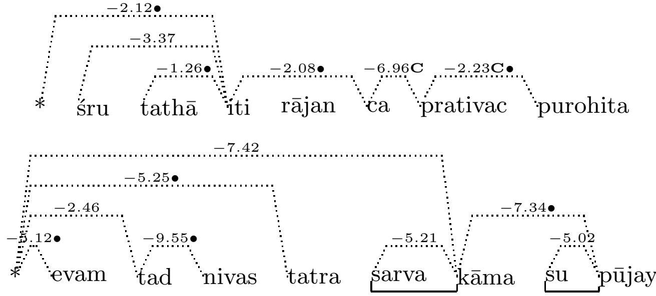


Fig. 2. Sample phrases parsed using all optimizations except for fixed syntactic rules – • = syntactic enforcement, **C** = probability estimated by smoothing

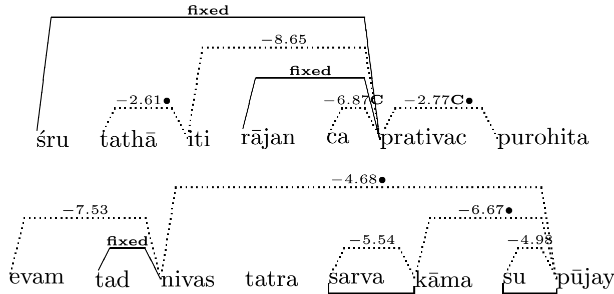


Fig. 3. Sample phrases parsed using all optimizations – Symbols are explained in the caption of Figure 2

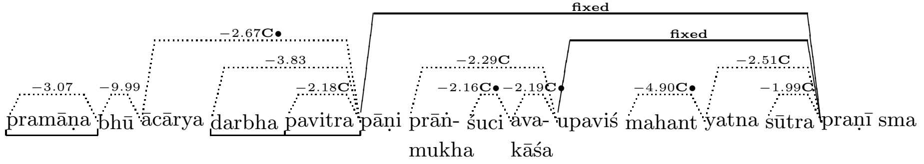


Fig. 4. "Benchmark sentence" parsed with all optimizations activated

3 Summary

In spite of the appealingly simple idea on which it is based, Yuret's parser is not able to correctly identify the syntactic structures of Sanskrit sentences. This behavior may be due to lack of training data. The performance of the parser can be improved when additional, non-lexical information about grammar and valences is included in the parsing process. The best performance is achieved when lexical and grammatical information is combined with a small number of fixed rules. These rules describe the basic components and structures of a complete sentence, but they are by far less detailed than the finite automata used by some researchers. Judging from the few samples that we have discussed in Section 2.3, such a hybrid approach can certainly be used as a starting point for building a database of the syntactic structures of classical Sanskrit. The strict projectivity of the dependency tree assumed in Yuret's original version of the algorithm remains an unsolved problem especially in the context of versified Sanskrit. In a future version of the parser, one may allow crossing links in the dependency structure if, for example, both links have a very high mutual information.

References

1. Brants, T.: TnT - a statistical part-of-speech tagger. In: Proceedings of the 6th Applied NLP Conference, Seattle (2000)
2. Canedo, J.: Zur Wort- und Satzstellung in der alt- und mittelindischen Prosa. Vandenhoeck & Ruprecht, Göttingen (1937)
3. Dagan, I., Lee, L., Pereira, F.C.N.: Similarity-based models of word cooccurrence probabilities. Machine Learning 34(1-3), 43–69 (1999)
4. Delbrück, B.: Altindische Syntax. Verlag der Buchhandlung des Waisenhauses, Halle (1988)
5. Gonda, J.: Old Indian. Handbuch der Orientalistik, Zweite Abteilung, Erster Band, Erster Abschnitt. E.J. Brill, Leiden (1971)
6. Goyal, P., Arora, V., Behera, L.: Analysis of Sanskrit text: Parsing and semantic relation. In: Proceedings of the First International Sanskrit Computational Linguistics Symposium, pp. 23–36 (2007)
7. Hellwig, O.: SanskritTagger, a stochastic lexical and POS tagger for Sanskrit. In: Proceedings of the First International Sanskrit Computational Linguistics Symposium, Rocquencourt, pp. 37–46 (2007)
8. Ickler, I.: Untersuchungen zur Wortstellung und Syntax der Chāndogyopaniṣad. Göppinger Akademische Beiträge, 75. Verlag Alfred Kümmerle, Göppingen (1973)

9. Speyer, J.S.: Vedische und Sanskrit-Syntax. Grundriss der Indo-arischen Philologie und Altertumskunde, III. Band, Heft A. Verlag von Karl J. Trübner, Strassburg (1896)
10. Staal, J.F.: Word Order in Sanskrit and Universal Grammar. Foundations of Language, Supplementary Series, vol. 5. D. Reidel Publishing Company, Dordrecht (1967)
11. Wauschkuhn, O.: Automatische Extraktion von Verbvalenzen aus deutschen Textkorpora. Shaker Verlag, Aachen (1999)
12. Yuret, D.: Discovery of Linguistic Relations Using Lexical Attraction. PhD thesis, Massachusetts Institute of Technology (1998)

Sanskrit Analysis System (SAS)

Manji Bhadra[1], Surjit Kumar Singh[2], Sachin Kumar[3], Subash[4], Muktanand Agrawal[5], R. Chandrasekhar[6], Sudhir K. Mishra[7], and Girish Nath Jha[8]

[1,2,3,5,8] Special Centre for Sanskrit Studies
Jawaharlal Nehru University
New-Delhi
[4] C-DAC Kolkata, NLP Group
[6] Visiting Scholar, Department of Classics, Brown University
[7] C-DAC Pune, AAI Group
[1]manji.bhadra@gmail.com, [2]surjit.jnu@gmail.com,
[3]sachinjnu@gmail.com, [8]girishjha@gmail.com,
[4]subhash.jnu@gmail.com, [5]mukta.jnu@gmail.com,
[6]chandrashekhara@gmail.com, [7]sudhirkumarmishra@gmail.com

Abstract. The paper describes Sanskrit Analysis System (SAS) – a complete analysis system for Sanskrit. Some modules of this system have already been developed. The system accepts full text inputs in Devanāgarī Unicode (UTF-8). The sandhi module does the segmenting for complex tokens and then hands over the text for detailed processing. Currently, the SAS modules have independent interfaces for unit testing at *http://sanskrit.jnu.ac.in*. The authors are working on the integration process of these modules. The SAS has two major components - the shallow parser and the kāraka analyzer. The shallow parser has separate modules, some of them are partially implemented, and some of them are in the process of being implemented. The modules have been developed as java servlet in unicode using RDMBS techniques. The applications of the SAS will be many ranging from being a Sanskrit reading assistant to machine translation system from Sanskrit to other languages.

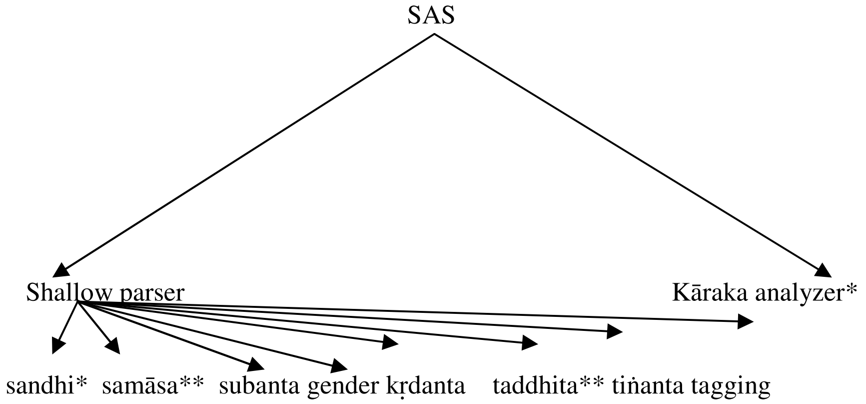
Keywords: sandhi, subanta, tiṅanta, kṛdanta, samāsa, taddhita, strī pratyaya, kāraka, vibhakti, prātipadika, dhātu, sup, tiṅ, avyaya, sūtra, vārttika, ākāṅkṣā, yogyatā, vivakṣā, liṅga, upadhā, gaṇa, pada, lakāra, vacana, vikāra, upasarga, vṛddhi.

1 Introduction

Developing an NLP system which analyzes a natural language is a difficult task. Sanskrit language has Pāṇini's grammar which is explicitly generative, while the task in analysis systems is to apply rules for processing a generated string or utterance. The authors in this process have tried to use Pāṇinian rules in reverse with appropriate lexical interfacing for analyzing Sanskrit. But in many cases, Pāṇinian rules demand

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 116–133, 2009.

deep semantic analysis, especially in the case of *kāraka* rules. The SAS has two major components - one is the shallow parser and the other is the *kāraka* analyzer. Shallow parser contains *sandhi*[1] analyzer, *samāsa* analyzer, *subanta*[2] analyzer, gender analyzer[3], *kṛdanta* analyzer[4], *taddhita* analyzer, *tiṅanta*[5] analyzer and the POS tagger[6]. Among them the *sandhi* analyzer is partially implemented and *samāsa* and *taddhita* analyzers are yet to be implemented. After getting the input, the system first analyzes the *sandhi* and *samāsa* by these modules wrapped into a separate system. Then the system analyzes the nominal inflected words for separating base and *vibhakti* and stores the PNG features of the tokens. Primary derived nouns are analyzed by the *kṛdanta* analyzer and secondary derived nouns are supposed to be analyzed by the *taddhita* analyzer. The *tiṅanta* analyzer analyzes verbs into affixes, number and person. After morphological analysis of each word in the sentence, the POS tagger assigns them appropriate POS category with the help of lexicon, corpus and other Pāṇini based formulations. The *kāraka*[7] analyzer takes over from here and analyzes the syntactico-semantic relations at the sentence level. A brief modular outline of the SAS is given below –



1 Kumar Sachin,2007, '*Sandhi Splitter and Analyzer for Sanskrit (with reference to ac Sandhi)*', M.Phil dissertation, Special Center for Sanskrit Studies, JNU.

2 Chandra Subash, 2006, '*Machine recognition and Morphological Analysis of Subanta-padas*', M.Phil dissertation, SCSS, JNU.

3 Bhadra Manji, 2007, '*Computational analysis of Gender in Sanskrit Noun Phrases for Machine Translation*', M.Mhil dissertation, Special Center for Sanskrit Studies, JNU.

4 Singh Surjit Kumar, 2008,'*Kṛdanta Recognition and Processing for Sanskrit*', M.Mhil dissertation, Special Center for Sanskrit Studies, JNU.

5 Agrawal Muktanada, 2007, '*Computational Identification and Analysis of Sanskrit Verb Forms of bhvādigaṇa*,' M.Phil dissertation, Special Center for Sanskrit Studies, JNU.

6 Chandrasekhar R, 2007, ' *Part of Speech Tagging for Sanskrit*', Ph.D. thesis, Special Center for Sanskrit Studies, JNU.

7 Jha Girish Nath, Misra Sudhir, '*Semantic processing in Pāṇini's karaka system*' Presented in second International Sanskrit Computational Linguistics Symposium at Brown University, 2008.

For example –

INPUT : भोजनान्तरं * धीमती किन्तु चञ्चला बालिका भ्रमित्वा धातुपाठं पठति ।
PREPROCESSING: भोजनान्तरं   धीमती किन्तु चञ्चला बालिका भ्रमित्वा धातुपाठं पठति ।
*SANDHI : भोजन अन्तरं
धीमती किन्तु चञ्चला बालिका भ्रमित्वा धातुपाठं पठति ।
**SAMASA: भोजन अन्तरं
धीमती किन्तु चञ्चला बालिका भ्रमित्वा
धातुणाम् पाठम्
पठति ।
SUBANTA: भोजन अन्तरं
धीमती् सु प्रथमा एकवचन
किन्तु[AV]
चञ्चला सु प्रथमा एकवचन
बालिका सु प्रथमा एकवचन
भ्रमित्वा
धातु ङस् षष्ठी वहुवचन पाठ अम् द्वितीया एकवचन
पठति[Verb]
GENDER: भोजन अन्तरं
धीमती[sf:hf] सु प्रथमा एकवचन
किन्तु[AV]
चञ्चला[sf:hf][टाप्]सु प्रथमा एकवचन
बालिका[sf:hf] [टाप्] सु प्रथमा एकवचन
भ्रमित्वा
धातु [sm:hm]ङस् षष्ठी वहुवचन पाठ[sm:hm] अम् द्वितीया एकवचन
पठति[Verb]> feminine
KṚDANTA: भोजन अन्तरं
धीमती [sf:hf] सु प्रथमा एकवचन
किन्तु[AV]
चञ्चला[sf:hf][टाप्]सु प्रथमा एकवचन
बालिका[sf:hf] [टाप्] सु प्रथमा एकवचन
भ्रमित्वा[भ्रम् त्वाच्]
धातु [sm:hm]ङस् षष्ठी वहुवचन पाठ[sm:hm] अम् द्वितीया एकवचन
पठति[Verb]> feminine
**TADDHITA: भोजन अन्तरं
धीमती [धीमत् मतुप्] [sf:hf] सु प्रथमा एकवचन
किन्तु[AV] चञ्चला[sf:hf][टाप्]सु प्रथमा एकवचन
बालिका[sf:hf] [टाप्] सु प्रथमा एकवचन

भ्रमित्वा[भ्रम् त्वाच्]
धातु [sm:hm]ङस् षष्ठी वहुवचन पाठ[sm:hm] अम् द्वितीया एकवचन
पठति[Verb]> feminine
TIṄANTA: भोजन अन्तरं
धीमती [धीमत् मतुप्] [sf:hf] सु प्रथमा एकवचन
किन्तु[AV]
चञ्चला[sf:hf][टाप्]सु प्रथमा एकवचन
बालिका[sf:hf] [टाप्] सु प्रथमा एकवचन
भ्रमित्वा[भ्रम् त्वाच्]
धातु [sm:hm]ङस् षष्ठी वहुवचन पाठ[sm:hm] अम् द्वितीया एकवचन
पठति { (कर्तृवाच्य) पठ ([भ्वादिगण] [सेट्] [सकर्मक]) ([लट्]) तिप् ([परस्मै] [प्रथम–पुरुष] [एकवचन]) } > feminine
POS TAGGER: भोजन अन्तरं[N]
धीमती[Adj][धीमत् मतुप्] [sf:hf] सु प्रथमा एकवचन
किन्तु[AV]
चञ्चला[sf:hf][Adj][टाप्]सु प्रथमा एकवचन
बालिका[N][sf:hf][टाप्]सु प्रथमा एकवचन
भ्रमित्वा[भ्रम् त्वाच्]
धातु[N][sm:hm]ङस् षष्ठी वहुवचन पाठ[N][sm:hm] अम् द्वितीया एकवचन
पठति { (कर्तृवाच्य) पठ ([भ्वादिगण] [सेट्] [सकर्मक]) ([लट्]) तिप् ([परस्मै] [प्रथम–पुरुष] [एकवचन]) } > feminine
KĀRAKA: भोजन अन्तरं](कर्म)[N]
धीमती(कर्ता)[Adj][धीमत् मतुप्] [sf:hf] सु प्रथमा एकवचन
किन्तु[AV]
चञ्चला(कर्ता)[Adj][sf:hf][टाप्]सु प्रथमा एकवचन
बालिका(कर्ता)[N][sf:hf] [टाप्] सु प्रथमा एकवचन
भ्रमित्वा[भ्रम् त्वाच्]
धातु[N][sm:hm]ङस् षष्ठी वहुवचन पाठ(कर्म)[N][sm:hm] अम् द्वितीया एकवचन
पठति { (कर्तृवाच्य) पठ ([भ्वादिगण] [सेट्] [सकर्मक]) ([लट्]) तिप् ([परस्मै] [प्रथम–पुरुष] [एकवचन]) } > feminine

2 Description of Each Module

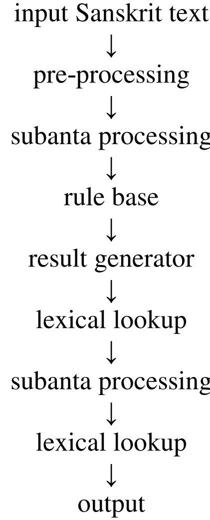
2.1 *Sandhi* Module

The analysis procedure of the *sandhi* analysis system uses lexical lookup method as well as rule base method. Before *sandhi* analysis process, pre-processing, lexical search of *sandhi* strings in *sandhi* example base and *subanta*-analysis takes place respectively. The pre- processing will mark the punctuation in the input. After that, the program checks the *sandhi* example base. This example base contains words of

sandhi-exceptions (*vārttika* list) and commonly-occurring *sandhi* strings (example list) with their split forms. These words are checked first to get their split forms without parsing each word for processing. After lexical search, *subanta* analyzer gets the case terminations (*vibhakti*) separated from the base word (*prātipadika*). *Subanta analyzer* also has a function to look into lexicon for verb and *avyaya* words to exclude them from *subanta* and *sandhi* processing.

2.1.1 *Sandhi* Rule Base

The rule base has been built up in the following format:



Rules for vowel sandhi are in format of

एऽ=ए+अ:(पूर्वरूपसन्धिः,एङःपदान्तादति);ओऽ=ओ+अ:(पूर्वरूपसन्धिः,एङःपदान्तादति);आय्s=ऐ+s:(अयादिसन्धि,एचोऽयवायावः);अय्s=ए+s:(अयादिसन्धि,एचोऽयवायावः);आव्s=औ+s:(अयादिसन्धि,एचोऽयवायावः);अव्s=ओ+s:(अयादिसन्धि,एचोऽयवायावः);?य्s=?ई+s:(यण् सन्धि इको यणचि);?य्s=?इ+s:(यण् सन्धि इको यणचि);

In these rules, the Roman character 's' stands for *svara* or vowel. This rule applies on the string after phoneme splitting. When phonemes are split, there are only vowels, consonants, *avagraha*, *visarga* and *anusvara*. For example the rule *आय्s=ऐ+s* means when in the phonemic string a sequence of characters appears as 'आ' followed by 'य्' then '*s*' (or any '*svara*'), then replace it by the right hand side of the '=' sign of the rule. In the RHS of the rule, '*s*' means that *svara* (not any *svara*) which is in LHS of the rule. The case of variable '*s*' is the same as in the rules of *ayādi sandhi*. Some of vowel *sandhi* rules make changes depending upon consonants. Operations depend upon consonants in following ways - on voiced consonants, unvoiced consonants and also as semivowels.

3 *Subanta* Analyzer

Sanskrit is a heavily inflected language, and depends on nominal and verbal inflections for communication for meaning. A fully inflected unit is called *pada*. Inflected nouns are called *subanta pada* and inflected verbs are called *tiṅanta pada*. According to Cardona[8], a Sanskrit sentence has NPs (including *avyayas* (AVs)) and VPs. It is defined as (N-En)p…(V-En)p. After *sup* and *tiṅ* combine with *prātipadika* (PDK)[9] they are assigned *kāraka* stipulations to return complete sentence.

3.1 Sanskrit *Subanta* (Inflected Nouns)[10]

Sanskrit nouns are inflected with seven cases in three markers. Sanskrit nouns can be further divided as primary derived forms (*kṛdanta*), secondary derived forms (*taddhita*), compounds (*samāsa*). There are 21 suffixes called *sup* (seven *vibhaktis* combined with three numbers)[11] which can be attached to the nominal bases (PDK) according to the syntactic category, gender and end-character of the base. Apart from these suffixes, there are *upasarga* (prefixes) which can attach to the PDK. But a PDK with only *upasarga* cannot be used in sentence without *vibhakti*. In Sanskrit, there are indeclinable (AVs) which are *subanta* but remain unchanged under all morphological conditions.[12]

3.1.1 Recognition of Punctuation

The system recognizes punctuations and tags them with the label _PUNCT. If the input has any extraneous character, then some normalization takes place. For example - UÉ/&^%@#म्:, बा,’”:–=लक: ➔ राम:, बालक: .
The Devanāgarī Sanskrit input text is then sent to the analyzer.

3.1.2 Recognition of *Avyaya*

The system takes help of *avyaya* database for recognizing AVs. If an input word is found in the AVs database, it is labeled AV, and excluded from the *subanta* analysis. Around 524 *avyayas* are stored in the database.

3.1.3 Recognition of Verbs

System takes the help of verb database for verb recognition. If an input is found in the verb database, it is labeled VERB and not sent for further processing. Since storing all Sanskrit verb forms is not a good option for computational reasons (there are 2000 verb roots and forms generated from it would be in the millions. Besides, there are innumerable *nāmdhātu*s as well and a regular verb form can be conjugated as

---

8 George Cardona, 1988, *Pāṇini His Work and Tradition*, vol…I Delhi(MLBD 1988).

9 अर्थवदधातुरप्रत्ययः प्रातिपदिकम् 1.2.45, कृत्तद्धितसमासश्च 1.2.46.

10 Jha Girish Nath et al., ‘*Inflectional Morphology Analyzer for Sanskrit*, pages 47-66 Proceedings of First International Sanskrit Computational Linguistics Symposium October, 2007.

11 स्वौजसमौट्छ ाभ्याम्भिस्ङेभ्याम्भ्यास्ङसिभ्याम्भ्यास्ङसोसांड्योस्सुप्.

12 सदृशं त्रिषु लिङ्गेषु सर्वासु च विभक्तिषु/ वचनेषु च सर्वेषु यन्न व्येति तदव्ययम् (Gopatha Brāhmaṇa).

sannata, *ṇijanata* etc as well). The SAS has 450 commonly used verb roots and their regular forms plus mechanisms to recognize unseen verbs (in the *tiṅanta* module) as well.

3.1.4 Recognition of *Subanta*

Thus a process of exclusion identifies the nouns in a Sanskrit text. After the punctuation, *avyayas* and verbs are identified, the remaining words in the text are labeled as SUBANTA.

3.2 Analysis of *Subanta*

System does analysis of inflected nouns with the help of two relational databases – examples and rules. Brief description of these databases follows-

3.2.1 Example Database

All complicated forms including those of some pronouns which cannot be easily analyzed according to rules are stored in the database. For example: अहम्=अस्मद्+सु प्रथमा एकवचन;अहं=अस्मद्+सु प्रथमा एकवचन; आवाम्=अस्मद्+औ प्रथमा द्विवचन; आवां=अस्मद्+औ प्रथमा द्विवचन;वयम्=अस्मद्+जस् प्रथमा वहुवचन;वयं=अस्मद्+जस् प्रथमा वहुवचन;

3.2.2 Rule Database

The *subanta* patterns are stored in this database. This database analyzes those nouns which match a particular pattern from the rule base. For example, रामः ,नदी, रमा, पुस्तकम् etc. First, the system recognizes *vibhakti* as the end character of nouns. For example, **':'** is found in nominative singular like- रामः यामः सर्वः भरतः एकः . The system isolates ':' and searches for analysis in the *sup* rule base. In the case of nominative and accusative dual, PDK forms will be ending in 'ौ' . For example, रामौ, यामौ, सर्वौ, एकौ. The system isolates 'ौ' and searches for analysis by matching the rule database. The sample data is as follows-

ा=ा+सु प्रथमा एकवचन;ाभ्याम्=+भ्याम् तृतीया चतुर्थी पञ्चमी द्विवचन;ाभ्यां=+भ्याम् तृतीया चतुर्थी पञ्चमी द्विवचन;भ्याम्=+भ्याम् तृतीया चतुर्थी पञ्चमी द्विवचन;भ्यां=+भ्याम् तृतीया चतुर्थी पञ्चमी द्विवचन;ेभ्यः=+भ्यस् चतुर्थी पञ्चमी बहुवचन;भ्यः=+भ्यस् चतुर्थी पञ्चमी बहुवचन;

4 Gender Analyzer

After *subanta* analyzer one can get information about Sanskrit nouns. But still gender information is not fully analyzed by *subanta* analyzer. In Sanskrit, there is gender agreement between adjectives and noun. Though there is no gender agreement between verb and the agent like Hindi, but *kṛdanta* forms agree with agent in terms of gender in a sentence. If machine has to understand Sanskrit language then it needs to understand the gender also like any other grammatical category. In the absence of a correct gender analysis of Sanskrit NPs, the target language translations may be wrong.

4.1 Description of the Gender Analyzer

The gender analyzer gets each sentence as a token. Then it sends the token for pre processing. After pre processing, it finds the verb and *avyayas* using database and excludes them for further processing. If in the text there are multiple NPs with conjunct or comma, then it gets separated NP chunks separated by conjunct or comma. After that, the system takes help of *subanta* analyzer to obtain PDK. After obtaining PDK, the system takes the help of lexical resources to get gender information of nouns. If enough information about gender is not found then the system looks for rules. At the end, it suggests the collocational gender of a sentence with respect to a target Hindi sentence.

4.1.1 Rule Base for Gender Analyzer

The present *subanta* analyzer of Sanskrit analyses the Sanskrit words with *prātipadika* with *vibhakti* markers. Some *vibhakti* markers help to identify the gender of a word. For example, the word *narān* can be analyzed as masculine gender from the *vibhakti* marker *ān,* and the number of the word would be plural. If the word appears in the input with this particular *vibhakti,* then the gender recognition of the word would be easy. The problem with this method is that the particular word has to arrive in the input with this particular *vibhakti.* As a consequence of this step, there would be huge numbers of words whose gender would be unrecognized by the system.

4.1.2 Rules of *Liṅgānuśāsana*

Gender can be recognized from the last but one syllable of the word. The technical name of this category is *upadhā* (penultimate).[13]

upadhā	**clause**	**Gender**	**Example**	**Exception**
k, ṭ, ṇ, ṭh, n, p, bh, ma, y, r, ṣ,s	if the word ends in *a*	Masculine	*stabaka, ghaṭa,dīpa, bhanu* etc	*chibuka, lalāṭa,pāpa, ratna* etc
L	if the word ends in *a*	Neuter	*phala*	*tūla, upala* etc

Among these words, if some words are used as proper names then it would follow the gender of a person if the name is a mythical and famous one, for example *ambarīṣa.*

After this step, the gender of a large number of words would remain unrecognized. To handle this problem, another rule from *Liṅgānuśāsana* is implemented for gender analyzer. The rule depends on the last *varṇa* of the word. For example, if the word ends in *ṛ* (*pitṛ, bhrātṛ*), generally the gender of the word would be masculine. But there are exceptions to this rule, like the words *mātṛ, nanāndṛ* etc in the feminine gender.

After the application of the Pāṇinian rules, there are still a large number of Sanskrit words whose gender recognition is very difficult. For these kinds of words, the gender

13 अलोऽन्त्यात्पूर्व उपधा.

can be recognized from the last syllable of the word apart from the Pāṇinian rule. For example, if the ending syllable is *a* then the gender of the word would be masculine. If the ending syllable is *ā* then generally the gender of that word would be feminine. But there are exceptions to this rule as *viśapā, dārā, hāhā* etc.

5 *Kṛdanta* Analysis

All the verbal suffixes besides *tiṅ* are called *kṛt*. *kṛt* is a technical term of Pāṇinian grammar that covers a vast field, both structurally as well as semantically.[14] The primary nominal derivatives from the verb roots are *kṛdanta*. The *kṛt* suffixes are added to roots or their modified forms, to form nouns, adjectives and indeclinables, for example *kṛ - kāra, krtṛ, karaṇa, kurvat, kariṣyat, cakṛvas, kṛtvā, kartum*. These are called *kṛdantas* or primary derived nominal bases.[15]

5.1 *Kṛdanta* Identification and Analysis Mechanisms

The process of *kṛdanta* analysis mechanism is divided into two sections - recognition and analysis.

5.1.1 *Kṛdanta* Identification Mechanism

The *kṛdanta* recognition starts by an exclusion process. The verb forms, *avyayas* and punctuations are excluded by running POS tagger by checking the verb, *avyaya* and pronoun databases and punctuation lists. The nominal bases are obtained by the *subanta* analyzer which is a part of the POS tagger. These nominal bases are then checked in fixed lists by the POS tagger. This may result in some of the *subantas* being marked for *kṛdanta*. The remaining *subantas* are sent to the *kṛdanta* recognizer and analyzer system for recognition and analysis using following steps –

- check the *kṛdanta* database, annotated corpus and *kṛdanta*-tagged Monier Williams Sanskrit Digital Dictionary (MWSDD).
- the *subantas* still untagged for *kṛdanta* are sent to the rule base for kṛdanta checking.
- the rule base applies *Pāṇinian* rule base in reverse for marking *kṛdantas*.
- it is possible that even after these systematic identification procedures, there may remain an untagged *kṛdanta subanta*. This will count as failure of the system.

5.1.2 *Kṛdanta* Analysis Mechanism

The system is divided into two parts- lexical database and rule-base. Lexical database of examples has been created for analyzing those forms which would be otherwise very complex to analyze if passed through the rule base. Lexical database has three major parts- a lexical *kṛdanta* database with complicated *kṛdanta* forms and their

---

14 Sharma, Dipti, *Structure and Meaning*,1982 Nag Publishers New-Delhi.
15 Kale, M.R., *A Higher Sanskrit Grammar.*

lexical information, Monier Williams Sanskrit Digital Dictionary and corpus of the current Sanskrit prose with *kṛdanta* words tagged with *kṛdanta* information.

The rule-base is for analyzing more regular forms. It consists of mainly three tables, namely, *upasargavikāra* table, *dhātuvikāra* table and *pratyayavikāra* table. To restrict *dhātuvikāra* and *pratyayavikāra* from inconsistent combinations, both are bound with a unique id.

For example, पाठकः[(पठ+ण्वुल्/पठ+णिच्+ण्वुल्)प्रथमा–एकवचन]

6 *Tiṅanta* Analysis

Verbs have been of central importance to Sanskrit grammarians. Yāska insisted so much on them that he propounded that all the nominal words are derived from verb roots[16]. Like noun *padas,* verb *padas* (*tiṅanta*) have to undergo certain inflectional process in which various verbal affixes are added to verb roots or *dhātus*. These *dhatus* are encoded with the core meaning of the verb. These can be primitive[17] or derived[18]. Primitive verb-roots, which are around 2000 in number, have been listed in a lexicon named *dhatupātha*. They are divided in 10 groups/classes called *gaṇas*. All the verb-roots of a group undergo somewhat similar inflectional process. Derived verb-roots may be derived from primitive verb-roots or from nominal forms. Prefixes also play an important role as they can change the meaning of a verb root. These roots then have to undergo various inflectional suffixes that represent different paradigms. In this process, the base or root also gets changed.

6.1 Process of Formation of Sanskrit Verb Forms

A Sanskrit verb root may take various forms. There are ten lakāras that represent Tense, Aspect and Mood. Inflectional terminations are 18 in number. These are divided in two groups – *parasmaipada* and *ātmanepada*, each having 9 affixes which is a combination of 3 persons x 3 numbers. A verb is conjugated in either *pada*, though some of the roots are conjugated in both. For each different *lakāra*, a root is affixed with these 9 terminations. Again, there are three voices- Active, Passive and Impersonal. Transitive verbs are used in the Active and Passive voices while intransitive verbs are conjugated in the Active and Impersonal voices. Addition of one or more of 22 prefixes (*upasargas*) to verb roots can result in more variety of forms. Derivative verb roots, both derived from verb roots as well as nominal words, also follow the same process to form verb forms. There can be some specific rules and exceptions in some cases. The following chart gives a rough estimate of possible verb-forms in Sanskrit[19] . This is to suggest that Sanskrit verb forms can not be stored in the database because the derived verb forms can be potentially innumerable.

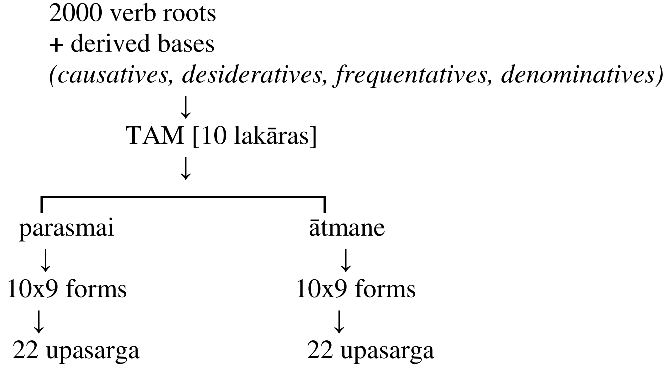
---

16 *bhāvapradhānamākhyātam* (Yāska, *Nirukta*).

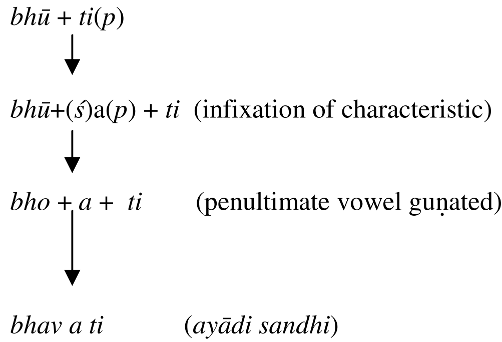
17 *bhuvādayo dhātavaḥ* (*P* 1/3/1).

18 *sanādyanta dhātavaḥ* (*P* 3/1/32).

19 Mishra Sudhir K., Jha, Girish N., 2004, *Identifying Verb Inflections in Sanskrit morphology*, in proc.of SIMPLE 04, IIT Kharagpur, pp. 79-81.



The verb roots of different *gaṇas* adapt certain terminations when *tiṅ* affixes are added to them. The *tiṇ* affixation also influences the verb root and it undergoes several morpho-phonemic changes, for example, having *guṇa* operation on the end vowel. The verb root can adopt certain operations resulting in the final verb-forms.



As shown in the example, when suffix *tip* is added to the verb root *bhū*, then *bhavati* form is obtained as the final verb form. This can be cited as a common analysis of most verb forms.

6.2 The Analysis of Sanskrit Verb Forms

The methodology for the analysis of Sanskrit verb form in the present work follows the analysis of Pāṇini in somewhat reverse direction. Pāṇinian analysis identifies different morphemes in any given *pada* and presents an analysis where he provides step-by-step methodology to derive a verb form from a given verb root in certain paradigms. As illustrated above, Sanskrit verb forms are a blend of multiple morphemes which contain relevant information. Analytically it can be said that the first element is the conjugational affix that remains at the end of every verb form. These affixes have encoded information of *pada* (though it is determined by root), lakāra, person and number. Thus termination can serve as the most important thing to convey about the paradigm information of any verb form. They can be a tool to identify a verb form in a given text. The terminations, as they are basically replacements of 18 original

tiṅ affixes in different *lakāras*, differ among themselves according to *lakāras*. However in each *lakāra,* they are similar for all verb roots of various groups, leaving some expectation. So *ti* can be used to identify any verb form of present tense of *parasmaipada.* But some terminations can vary among themselves for a group of *gaṇas*. Then again, the termination may be changed due to morphophonemic environment, *tā* affix of *luṭ lakāra,* changing to *ṭā* with roots like *yaj*.

Further left, there are various morphemes of the various characteristics and increments inserted between the verb root and terminations, in the process of their formation explained above. *Bhvādigaṇa* verb forms in conjugational *lakāras*, have '*a*' as a result of *śap* characteristics; *svādi* roots have *no*, *nu* or *nv* all of them remaining morphemes of *śnu*. Some roots like that of *adādi* have no such characteristics sign infixed in them.

At the right end of the verb form, there is modified stem of the verb root. The modification can be *guṇa*, *vṛddhi* or any other. Generally a root adopts a common stem in all the forms for both *padas* in conjugational *lakāras*. So *bhav* is the stem for all *parasmaipadī* forms in the conjugational *lakāras*. But there are exceptions to it to that extent that four or five types can be found among nine forms of a single *lakāra pada*.

Here the first morpheme the *tiṇ* termination is common among all verb forms of a particular *pada-lakāra- puruṣa -saṁkhyā* combination. Second constituent, the characteristics (existing in the form of its remaining morpheme) and increments inserted in between may differ, yet being almost the same in a particular group. The third constituent, the modified verb-root is particular in the strict sense. In the analysis, the recognition of the *tiṅ* will identify a word as a verb form and find out its *pada-lakāra- puruṣa -saṁkhyā*. The second morpheme can, in many cases, be helpful to recognize the *gaṇa* of a particular root because the characteristics in a *lakāra* are determined by the *gaṇa* that the roots belong to. Thus the core of the analytical approach is that each *tiṅanta* verb form can be analyzed to form a unique combination of verbal stem + *tiṅ* termination, and both of these constituent parts are stored in separate tables. When it is to be analyzed, its constituent morphemes are recognized and identified with the help of pre-stored structured data.

7 POS Tagger

After getting the information about inflected nouns of Sanskrit, it is necessary to understand the role of each word in a sentence. This process of marking up the words in a text as corresponding to a particular part of speech, based on both its definition, as well as its context—i.e., relationship with adjacent and related words in a phrase, sentence, or paragraph[20] is called POS tagging. A typical POS tagger acts as a shallow parser and is pre- requisite in several NLP related applications such as machine translation system, information retrieval word sense disambiguation etc. Sanskrit is an inflectional language and words in a sentence carry information about entities in terms of stem, endings, gender, case, number and case relation, while verbs denote activity, function reaction, mode, voice, tense, person, number etc. Extracting and

[20] http://en.wikipedia.org/wiki/Part-of-speech_tagging

organizing, i.e. annotating, these information is the first step towards understanding the language. Words in a language may occur in POS or various grammatical categories as they are also known. In Sanskrit for example

1) *gacchati* can be either a *tiṅanta* or *kṛdanta*
2) *rāmaḥ* can either be a *nāmapada* (*abhidhāna*) or *tiṅanta*
3) *āyātaḥ* can either be a *kṛdanta* or *tiṅanta*
4) *mā* can either be an *avyaya* or *namapada* or a *sarvanāman* etc

7.1 The Sanskrit Tagset

The designed tagset is classified according to the morphological structure of the categories. There are two kinds of tags in this tagset. Word class main tags and feature sub-tags. The tag as a whole is a combination of word class main tag with feature sub-tags separated by an underscore. All the tags bear Sanskrit names[21] with letter-digit acronymic in Roman script.

The process first involved evolving a stable tagset for Sanskrit text which has 65 word class tags, 43 feature sub-tags, and 25 punctuation tags and one tag *UN* to tag unknown words – a total of 134 tags. A single full tag is a combination of word class tag and feature sub-tags (indeclinable and punctuation tags do not have sub-tags). The word class tags are 8 Noun tags, 8 Pronoun tags, 3 Adjective tags, 9 Participle tags, 2 Number tags, 14 Compound tags, 11 indeclinable tags and 10 verb tags. Feature tags are three gender sub tags (p,s,n); 8x3 = 24 (Nominal)Case and Number tags (1.1 through 8.3); 4 Verb base modifying tags (Nd, Yn, Sn, Ni); 1Verbal Preposition (UPA); 2 Pada tags (P and A); 3x3 = 9 (Verbal) Person and Number tags (1.1 through 3.3).

7.2 Description POS Tagger

7.2.1 Pre-processing

After getting Unicode (UTF-8) sandhi free Devanagari Sanskrit input (or with minimal sandhi) as word, sentence or text, the system sends those input for pre processing. In this step, the system searches for punctuations in the input and tags them. In addition to tagging the punctuations, this function also removes unwanted foreign letters or punctuations from the inside of a Devanagari string.

7.2.2 Fixed-List Tagger

After initializing the input, the system goes to check in the fixed tagged lists. This database stores lists of *avyayas*, list of verbs and POS list. The POS example base consists of approximately 1 MB data. For example

अञ्चति[P_laTV_1.1]/[KV1_p_7.1]/[KV1_n_7.1];अञ्चतः[P_laTV_1.2]/[KV1_p_2.3]/[KV1_p_5.1]/[KV1_p_6.1]/KV1_n_5.1]/[KV1_n_6.1];अञ्चन्ति[P_laTV_1.3]/[K

[21] Few names are coined in English for the purpose of clarity and to avoid confusion while marking their notions. The tags having English names are all the compound tags containing 'C' for Compound and few punctuation tags.

V1_n_1.3]/[KV1_n_2.3];अञ्जति[P_laTV_1.1]/[KV1_p_7.1]/[KV1_n_7.1];अञ्जतः[P_laTV_1.2]/[KV1_p_2.3]/[KV1_p_5.1]/[KV1_p_6.1]/KV1_n_5.1]/[KV1_n_6.1];अञ्जन्ति[P_laTV_1.3]/[KV1_n_1.3]/[KV1_n_2.3];नाम[N_n_1.1]/[(nAman)N_n_2.1];प्रथमम्[N_n_1.1]/[(prathama)N_n_2.1]/[N_p_2.1]/[AVKV];तन्त्रम्[N_n_1.1]/[(tantra)N_n_2.1];यस्य[SNS_n_6.1]/[(yad)SNS_p_6.1];महान्[NVI_p_1.1]/[(maha)N_p_2.3];सिंहगोवृषयोः[N_p_6.2]/[(siMhagovRuSha)N_p_7.2]/[N_s_6.2]/[(siMhagovRuShA)N_s_7.2];वने[N_n_7.1]/[(vana)N_n_1.2]/[(vana)N_n_2.2];तत्[SNN_n_1.1]/[(tad)SNN_n_2.1];महिलारोप्यम्[N_n_1.1]/[(mahilAropya)N_n_2.1];नगरम्[N_n_1.1]/[(nagara)N_n_2.1];

If the token is found, it gets tagged with corresponding tag from the lexicon.

7.2.3 *Subanta* Analyzer

However, a large number of input tokens are not found in these lists as they may be marked for *subanta*. Therefore the next component of *subanta* analyzer checks untagged input in the *subanta* examples. If not found, it starts analyzing the token from the right end and checks in the lexicon after each appropriate cut. If it is found, it tags the input. If after all these steps, the input remains untagged, it gets the ‘not found’ tag. The resultant tagged token is sent back to the main tagger ‘Post’ which linearizes the results with adding color schemes for ambiguous and untagged tokens.

8 *Kāraka* Analysis

After understanding words in a Sanskrit sentence, it is necessary to understand how the words are arranged in a sentence, what are the relation between other words and verbs. In Sanskrit, this relation can be understood while analyzing *kāraka* relation. Etymologically *kāraka* is the name given to the relation between a noun and a verb in a sentence. It means ‘that which brings about’ or ‘doer’. [22]

8.1 *Kāraka* and *Vibhakti* Mapping

Pāṇini discusses the entire gamut of *kāraka-vibhakti* relations in three sections of *Aṣṭādhyāyī*

- *kāraka sūtra* (P. 1.4.23 – P. 1.4.55) → 33 *sūtras*
- *vibhakti sūtra* (P. 2.3.1 - P. 2.3.73) → 73 *sūtras*
- *karma-pravacanīya* (P. 1.4.82 – P. 1.4.97) →16 *sūtras*

Now the problem of implementation of all *kāraka* rules is that there are rules of *vivakṣā* dependent operations. In the example स्थाल्या पचति, स्थाली should be Location as it is the आधार (आधारोऽधिकरणम्), but it is करण by rule साधकतमं करणम्

[22] A detailed description of kāraka and its mapping with vibhakti is given in Jha Girish Nath, Mishra Sudhir K., ‘*Semantic processing in Pāṇini’s kāraka system*’ Presented in second International Sanskrit Computational Linguistics Symposium at Brown University, 2008.

because the speaker thinks it is the most instrumental (साधकतम (प्रकृ उपकारक)) and therefore prefers Instrumental case. कर्मणा यमभिप्रैति स सम्प्रदानम prescribes Dative for the receiver of gift, but vārtika ्अशिष्टव्यवहारे दाणः प्रयोगे चतुर्थ्यर्थे तृतीया prohibits it if the gift was intended for deriving some benefit (sexual favor in this case). Vārtikas extend, limit Pāṇinian rules, for example - नी–वह्योर्न (नाययति वाहयति वा भारं भृत्येन) allows karaṇa if the verb is नी or वह . It thus limits गतिबुद्धिप्रत्यवसानार्थ शब्दकर्माकर्मकाणामणि कर्ता स णौनियन्तृकर्तृकस्य वहेरनिषेधः which allows karma. Sometimes vārtikas limit themselves नियन्तृकर्तृकस्य वहेरनिषेधः (if the *kartā* is '*sārathi*' or any of its synonyms then नी–वह्योर्न does not apply → वाहयति रथं वाहान् सूतः (*karma* by Pāṇini's गतिबुद्धि… sūtra). Another problem is how to implement semantic conditions such as, स्वातंत्र्य,ईप्सित/ईप्सिततम्,साधकतम,अभिप्रैति (to be अभिमुख - approach someone for gift), प्रीयमाणः (one who gets pleased - रुच्यर्थानां प्रीयमाणः) – हरये रोचते भक्तिः,ध्रुव (fixed point) अपाय (path of separation) ध्रुवमपायेऽपादानम्,धावतः अश्वात् पतति (is अश्वात an आधार or ध्रुव ?) ,आधारोऽधिकरणम (आधारः किम् ?) etc.

A tentative model of *kāraka* analyzer is given below.

1. VERB ID
2. VERB ANALYSIS
3. NON—VERB ID
4. SUBANTA ANALYSIS
****5. ĀKĀṄKṢA CHECK**
****6. KĀRAKA RULES**
****7. SPECIAL CONDITIONS**
8. KĀRAKA ASSIGNMENT

In this model, the starred modules are under implementation. While analyzing the verb, the system will take the help of *tiṅanta* analysis. For tokenizing the *tiṅanta* , the system checks every character of the word through reverse module and matches through verb database for recognizing the *tiṅanta pada* which is used in the sentence. If it is found, then all information which is relevant in *kāraka* analysis are provided to system for further implementation. Otherwise it returns to check again if *dhātu* is used with *upasarga* and after recognizing *upasarga,* the system removes the *upasaga* from the verb, and again checks it for *dhātu* identification number and the result is sent to *dhātu* information database for getting the relevant information of the *dhātu*. After the verb analysis, the system checks for non verb words and then it takes help of *subanta* analyzer and *kāraka* assignment is implemented. In between, there are some steps like *ākāṅkṣā* checking, and special semantic conditions are not implemented yet.

9 Result Analysis and Limitations

Currently the modules of the SAS are not integrated. Individual modules can be tested as http://sanskrit.jnu.ac.in. The limitation of lexical resource may affect some

modules. In *sandhi* analyzer, if the input is हिमालयः the output will be हिमाले अः (अयादिसन्धि एचोऽयवायावः),हिमाली अः (यण् सन्धि इको यणचि), हिमालि अः (यण् सन्धि इको यणचि), हिमा अलयः (दीर्घसन्धि अकः सवर्णे दीर्घः), हिम अलयः (दीर्घसन्धि अकः सवर्णे दीर्घः) हिम आलयः (दीर्घसन्धि अकः सवर्णे दीर्घः). Here, the system gives multiple answers with appropriate rules of Pāṇini as it finds all the parts in these results as valid words in the 200k Sanskrit dictionary. Future enhancements in this module will select the most common output based on a frequency marling in the dictionary. The *subanta* analyzer can not recognize many forms and is being currently updated. In the gender analyzer, lexical resources may hamper the result. The system cannot identify gender of those words which are used in different gender in different meaning properly, like the word *mitra*. Sometimes the system fails to check proper gender agreement as well. There are limitations of the *Kṛdanta, Tiṅanta*, POS tagger and *Kāraka* modules as well which are being improved currently.

10 Conclusion

The authors in this paper have presented an ongoing work for developing a complete SAS. Currently, the SAS has some modules partially developed and some under development. Significant future additions will be the ambiguity resolution modules like anaphora resolution. After the *kāraka* checking module, a disambiguation module is also going to be added in near future, to resolve problems like '*bhavati ! bhikṣām dehi*'. Here according to the SAS system, *bhavati* and *dehi* both get verb tags. But here *bhavati* is used as noun and in vocative. If there is proper punctuation like an exclamation mark after this word then one can say it is used in vocative. If there is no punctuation mark then the problem can be resolved by counting verbs in the sentence which in most cases can be only one. These kinds of problems are to be handled in the disambiguation module. For the testing of the system, 140 files in unicode Devanāgari have been collected. Those texts are in simple Sanskrit and collected from different sources mostly samples of current Sanskrit. Though there is no complete statistics of the results, but one of the tests in *subanta* with simple Sanskrit gave a 90% accuracy.

The table is given below.

S.No.	File	Theme	Source	Words	Time(secs)
1	Corpus-1	*rājā sagaraḥ*	*sandeśaḥ*	609	3
2	Corpus-2	*samrāṭa aśokaḥ*	*sandeśaḥ*	916	3.2
3	Corpus-3	*ekaḥ nibandhaḥ*	*sandeśaḥ*	882	3
4	Corpus-4	*cācā neharuḥ*	*sandeśaḥ*	332	1
5	Corpus-5	*sarasvatī vandanā* and a story	*sandeśaḥ*	241	1
6	Corpus-6	*ādhunika praśāsanaḥ*	*sandeśaḥ*	1045	3.5
7	Corpus-7	*ekaḥ vaṇikaḥ*	*sandeśaḥ*	849	2
8	Corpus-8	*paśya me rūpāṇi*	*sandeśaḥ*	1328	4
9	Corpus-9	Sanskrit *sikśā*	*sandeśaḥ*	306	2
10	Corpus-10	*saṅghe śaktiḥ*	*sandeśaḥ*	4207	6

References

1. Vamdeva, A.: 'Liṅga-Parijnānam', Shabdatattva Prakāshan Varanasi (1990)
2. Muktanada, A.: Computational Identification and Analysis of Sanskrit Verb Forms of bhvādigaöa. Mphil degree at SCSS, JNU (submitted, 2007)
3. Manji, B.: Computational analysis of Gender in Sanskrit Noun Phrases for Machine Translation. Mphil degree at SCSS, JNU (submitted, 2007)
4. Bharati, A., Chaitanya, V., Sangal, R.: A Computational Gram-mar for Indian Languages Processing. Indian Linguistics Journal 52, 91–103 (1991)
5. Bharati, A., Chaitanya, V., Sangal, R.: Natural Language Processing: A Pan Perspective. Prentice-Hall of India, New Delhi (1995)
6. George, C.: Pāṇini's syntactic categories. Journal of Oriental Institute Baroda 16, 201–215 (1967)
7. George, C.: Pāṇini His Work and Tradition (MLBD 1988), Delhi, vol. I (1988)
8. George, C.: Some Questions on Pāöini's Derivational System. In: Procs. of Splash, iSTRANS, p. 3. Tata Macgraw-Hill, New Delhi (2004)
9. Chandrasekhar, R.: Part of Speech Tagging for Sanskrit. Phd degree at SCSS, JNU (submitted, 2007)
10. Daniel, J., Martin, J.: Speech and Language Processing. Prentice-Hall of India, New Delhi (2000)
11. Edgren, A.H.: On the verbal roots of the Sanskrit language and of the Sanskrit grammarians. Journal of American Oriental Society 11, 1–5 (1885)
12. Huet, G.: Towards Computational Processing of Sanskrit, Recent Advances in Natural Language Processing. In: Proceedings of the International Conference ICON, Mysore, India (2003)
13. Jha, Girish N., et al.: Towards a Computational analysis system for Sanskrit. In: Proc. of first National symposium on Modeling and Shallow parsing of Indian Languages at Indian Institute of Technology, Bombay, pp. 25–34 (2006)
14. Jha, Girish N.: A Prolog Analyzer/Generator for Sanskrit Noun phrase Padas, Language in India, vol. 3 (2003)
15. Jha, Girish N.: Generating nominal inflectional morphology in Sanskrit. In: SIMPLE 2004, IIT-Kharagpur Lecture Compendium, pp. 20–23. Shyama Printing Works, Kharagpur (2004)
16. Jha, Girish N.: Morphology of Sanskrit Case Affixes: A computational analysis, M.Phil dissertation, J.N.U., New Delhi (submitted, 1993)
17. Jha, Girish N.: The system of Panini. Language in India 4, 2 (2004)
18. Jha, Girish N., Mishra, Sudhir K.: Semantic processing in Pāṇini's karaka system. In: Second International Sanskrit Computational Linguistics Symposium at Brown University (2008)
19. Joshi, S.D.: Verbs and nouns in Sanskrit. Indian linguistics 32, 60–63 (1962)
20. Kale, M.R.: A Higher Sanskrit Grammar. MLBD, New Delhi (1995)
21. Kapoor, K.: Semantic Structures and the Verb: a propositional analysis. Intellectual Publications, New Delhi (1985)
22. Sachin, K.: Sandhi Splitter and Analyzer for Sanskrit (with reference to ac Sandhi). Mphil degree at SCSS, JNU (submitted, 2007)
23. Mishra, Sudhir K., Jha, Girish N.: Identifying Verb Inflections in Sanskrit morphology. In: Proc. of SIMPLE 2004, IIT Kharagpur, pp. 79–81 (2004)
24. Mishra, Sudhir K., Jha, Girish N.: Sanskrit Karaka Analyzer for Machine Translation. In: SPLASH proc. of iSTRANS, pp. 224–225. Tata McGraw-Hill, New Delhi (2004)

25. Ruslan, M.: The Oxford Handbook of Computational Linguistics. Oxford University Press, Oxford
26. Mishra, N. (ed.): Kashika of Pt.Vamana and Jayaditya, Chaukhamba Sanskrit sansthan, Varanasi (1996)
27. Van Nooten, B.A.: Pāṇini's replacement technique and the active finite verb. University of California, Berkeley
28. Sharma, R.N.: The Aṣṭādhyayi of Pāṇini. Munshiram Manoharlal Publishers Pvt. Ltd, Delhi (2003)
29. Shastri, Bheemsen, Laghusiddhantakaumudi, Prakashan, B.: 537, Lajapatrai Market, New Delhi
30. Kumar, S.S.: Kṛdanta Recognition and Processing for Sanskrit. Mphil degree at SCSS, JNU (submitted, 2008)
31. Shastri, Dwarikadas, S.: The Madhaviya Dhatuvṛtti by Saya_acarya. Tara Book Agency, Varanasi (2000)
32. Sharma, D.: Structure and Meaning. Nag Publishers, New Delhi (1982)
33. Subash, Jha, Girish N.: Morphological analysis of nominal inflections in Sanskrit. In: Platinum Jubilee International Conference, L.S.I., p. 34. Hyderabad University, Hyderabad (2005)
34. Subash: Machine recognition and morphological analysis of Subanta-padas, M.Phil dissertation J.N.U., New Delhi (submitted, 2006)
35. Upadhye, P.V.: Dhaturupacandrika. Gopal Narayen & Co, Bombay (1927)
36. Whitney, W.D.: History of Sanskrit Grammar, Sanjay Prakashan, Delhi (2002)

Web References

- IIIT, Hyderabad, `http://www.iiit.net/ltrc/Publications/Techreports/tr010/anu00kbcs.txt`
- Peter M. Scharf and Malcolm D. Hyman, `http://sanskritlibrary.org/morph/`
- Huet's site http://sanskrit.inria.fr/
- Prajna system, ASR Melcote, `http://www.sanskritacademy.org/Achievements.htm`
- Aiba, Verb Analyzer for classical Sanskrit,
- `http://wwwasia.human.is.tohoku.ac.jp/demo/vasia/html/`
- Desika, TDIL, Govt. of India, `http://tdil.mit.gov.in/download/Desika.htm`
- RCILTS, JNU, http://rcilts.jnu.ac.in
- Shabdabodha, ASR, Melcote, `http://vedavid.org/ASR/#anchor2`
- `http://en.wikipedia.org/wiki/Part-of-speech_tagging`

Translation Divergence in English-Sanskrit-Hindi Language Pairs

Pawan Goyal[1] and R. Mahesh K. Sinha[2]

[1] School of Computing and Intelligent Systems, University of Ulster, UK
goyal-p@email.ulster.ac.uk
[2] Indian Institute of Technology, Kanpur, India
rmk@iitk.ac.in

Abstract. The development of a machine translation system needs that we identify the patterns of divergence between two languages. Though a number of MT developers have given attention to this problem, it is difficult to derive general strategies which can be used for any language pair. Therefore, further exploration is always needed to identify different sources of translation divergence in different pairs of translation languages. In this paper, we discuss translation pattern between English-Sanskrit and Hindi-Sanskrit of various constructions to identify the divergence in English-Sanskrit-Hindi language pairs. This will enable us to come up with strategies to handle these situations and coming up with correct translation. The base has been the classification of translation divergence presented by Dorr [Dorr, 1994].

Keywords: Machine Translation, Translation Divergence.

1 Introduction

Translation divergence occurs when the underlying concept of a sentence gets manifested differently in different languages. The topic has been studied from different perspectives and a number of approaches have been proposed to handle them [Habash and Dorr, 2002]. It is difficult to obtain correct machine translation for any MT system without identifying the nature of translation divergence. In this paper, we examine English-Sanskrit and Hindi-Sanskrit language pairs mostly from the perspective of identifying the language specific divergences. The languages, English and Sanskrit, as well as Hindi and Sanskrit differ in many respects, presenting a rich source for the study of translation divergence in MT. In section 2, we look at the translation divergence classification proposed by Dorr and in what way, it appears in the English-Sanskrit-Hindi language pairs. In section 3, we look at some other divergence patterns which cannot be classified in the divergence patterns identified by Dorr. In section 4, we give the concluding remarks.

2 Divergence Patterns Identified by Dorr

Dorr has identified seven classes of translation divergences. These classes are:

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 134–143, 2009.

1. Thematic Divergence
2. Promotional Divergence
3. Demotional Divergence
4. Structural Divergence
5. Conflational Divergence
6. Categorial Divergence
7. Lexical Divergence

These classes have been defined to account for different types of translation divergences found in a pair of translation languages. Let us look, in what sense these can be seen while we examine English-Sanskrit and Hindi-Sanskrit language pairs.

2.1 Thematic Divergence

Thematic divergence refers to the divergences arising from differences in the realization of the argument structure of a verb. In the language pairs, we have considered, we can find many examples of this divergence. Let us consider some of these:

– Subject NP in English in nominative case while subject NP in Sanskrit in dative case:

1. I like sweets. ⇒ *mahyam madhuram rocate.*
   (I DAT) (sweets) (like.PR[1]).
   ⇒ Sweets are liked by me.

When we go from Hindi to Sanskrit, same divergence appears:

2. *main miṭhāī pasanda karatā hūM.* ⇒*mahyam madhuram rocate.*
   (I) (sweets) (like) (do) (be.PR) (I DAT) (sweets) (like.PR).
   ⇒ *mujhe miṭhāī pasanda haiṃ.*
   (I DAT) (sweets) (like.PR).

From the example 2, we see that there is divergence between Hindi and Sanskrit. Here the experiencial verb '*ruc*' gets an active construction in Hindi while it conditions a dative subject in Sanskrit. However, there is no divergence when we go from Sanskrit to Hindi since the closest translation in Hindi is *mujhe miṭhāī pasanda haiṃ* which has a dative subject as well.

If we examine other verbs such as 'eat' for the same pattern, we will not find this divergence.

3. I eat sweets. ⇒*aham madhuram khādāmi.*
   (I) (sweets) (eat.PR)
   ⇒ *main miṭhāī khātā hūM*
   (I) (sweets) (eat.PR)

[1] Appendix 1.

Thus, this divergence is present in a special category of verbs and not all the verbs. *Pāṇini* in his *kāraka adhikāra* lists special cases of verbs which require special treatment. In a work on English verb classes by Levin [Levin, 1997], semantic classes of verbs are analyzed which give rise to Divergence.

2.2 Structural Divergence

These are the examples where a noun phrase (NP) is realized in different ways in two languages. This is most common between English and Sanskrit because in Sanskrit, no noun is pronounced without a *vibhakti*. This *vibhakti* can be realized in English either as a null or a preposition. Here are some of the examples that exhibit Structural divergence:

4. He brought mangoes.	⇔ *saḥ āmrāni ānayat.* (He) (mangoes) (bring.PST)
5. He went to the market.	⇔ *saḥ āpaṇam agacchat.* (He) (market.ACC) (go.PST)
6. He enters the class.	⇔ *saḥ kakṣāyām praviśati.* (He) (class.LOC) (enter.PR)

In example 4 and 6, the *vibhakti* in Sanskrit is not realized by a preposition in English but a null, while in example 5, the *vibhakti* is realized by a preposition in English.

While we go from Sanskrit to hindi, we will not get many examples of this divergence since both languages are *kāraka* and *vibhakti* based.

2.3 Conflational and Inflational Divergence

A conflational divergence results when two or more words in English are translated by one word in Sanskrit. There are many mechanisms in Sanskrit that present this divergence. Same is true between Hindi and Sanskrit too. Let us look at some of them:

- ‘***sannata prayoga***’:-

  7. *aham pipaṭhiṣāmi* ⇒ I want to read.
  (I) (want to read)
  ⇒ aham paṭhitum icchāmi
  (I) (to read) (want)
  aham pipaṭhiṣāmi ⇒ *main paḍhanā cāhatā hūM.*
  (I) (want to read) (I) (read) (want be.PR)
  ⇒ aham paṭhitum icchāmi
  (I) (to read) (want)

  A sentence in Sanskrit such as ‘*aham pipaṭhiṣāmi*’ is translated in English as ‘I want to read’, thus ‘want to see’ is translated as ‘*pipaṭhiṣāmi*’. Thus it

presents conflational divergence. This divergence is exhibited even between Hindi and Sanskrit.

– ‘***nāmadhātu prakriyā***’:- In certain meanings, a Sanskrit nominal stem can accept certain *pratyayas*. Consider the sentences:

8. *saḥ paṇḍitāyate* ⇒ He behaves like a scholar.
(He) (behaves like a scholar)
⇒ *saḥ paṇḍitāḥ iva ācarati*
(He) (Pandita) (like) (behave.PR)

saḥ paṇḍitāyate ⇒ *vaha paṇḍita kī taraha ācaraṇa karatā hai.*
(He) (behaves like a scholar) (He) (scholar) (like) (behave do be.PR)
⇒ *saḥ paṇḍitāḥ iva ācarati*
(He) (Pandita) (like) (behave.PR)

He behaves like a scholar.⇐ *saḥ paṇḍitāyate*. The affix *kyaṅ* is applied to the noun *paṇḍita* in the sense of ‘behaves like’.

9. *saḥ śiṣyam putrīyati* ⇒ He treats the disciple as his son.
(He) (disciple.ACC) (treats as son)
⇒ *saḥ śiṣyam putram iva ācarati*
(He) (disciple.ACC) (son) (like) (behave.PR)

saḥ śiṣyam putrīyati ⇒ He treats the disciple as his son.
The affix *kyac* is applied to the noun *putra* in the sense of ‘treats like’.

In example 9, ‘treats as son’ in English is realized by a single word ‘*putrīyati*’ in Sanskrit and presents an example of conflational divergence.

– ‘***yaṅanta prayoga***’:- This refers to frequentatives.

10. *saḥ pāpacyate* ⇒he cooks again and again
(He) (cooks again and again)
⇒ *saḥ punaḥ punaḥ pacati*
(He) (again) (again) (cook.PR)
Same pattern is exhibited when we go from Sanskrit to Hindi language pair.
saḥ pāpacyate ⇒ *vaha bāra bāra pakātā hai*
(He) (cooks again and again) (He) (again) (again) (cook be.PR)
⇒ *saḥ punaḥ punaḥ pacati*
(He) (again) (again) (cook.PR)

Hence the English words, ‘cooks again and again’ and the Hindi words, ‘*bāra bāra pakātā hai*’ will be translated in Sanskrit as ‘*pāpacyate*’. Here, the root ‘pac (to cook)’ is applied with the affix ‘*yaṅ*’ to form ‘*pāpacyate*’. This again, exhibits the example of conflational divergence.

2.4 Categorial Divergence

Categorial divergences are located in the mismatch between parts of speech of the pair of translation languages. Consider the following example:

11. She is jealous of me. ⇔ *sā mahyam īrṣyati*
(She) (with me) (jealousy does).

We notice that in Sanskrit, 'jealous' is realized by a verbal mapping, thus presenting categorial divergence. When we go from Hindi to Sanskrit, we have another translation possible in hindi, for example:

usako mujhase īrṣyā hai ⇒ *sā mahyam īrṣyati*
(She.DAT) (me-from) (jealousy be.PR) (She) (with me) (jealousy does).
⇒ *vaha mujhase īrṣyā karatī hai*
(she) (me-with) (jealousy) (do)

2.5 Lexical Divergence

Lexical divergence arises out of the unavailability of an exact translation map for a construction in one language into another language. In Sanskrit, by adding '*upasarga*' to a verb, it gets a different meaning. For example, consider the following sentences:

12. *saḥ mām vadati* ⇒ He speaks to me.
(He) (me.ACC) (speaks)

13. *saḥ kṣetre vivadati* ⇒ He quarrels in the field.
(He) (field.LOC) (quarrel.PR)
⇒ *saḥ kṣetre kalahaṃ karoti*
(He) (field.LOC) (quarrel) (do)

In example 12, the Sanskrit verb '*vad*' is realized by English verb 'speak', while in example 13, the Sanskrit verb '*vi+vad*' (*upasarga 'vi'* is added to verb *vad*) is realized by a new verb in English 'quarrel'.

3 Other Divergence Patterns

Let us look at some other divergence patterns that are found in these languages:

3.1 Implications of Word Order

Though Sanskrit is a free phrase order language, there are situations where the word order changes the meaning of the sentence. An example is the occurrence of *kim*, consider the sentences:

14. *kim saḥ khādati?* ⇔ 'is he eating?'
(QP) (he) (eat.PR)

15. *saḥ kim khādati?* ⇔ 'what is he eating?'
(He) (IP) (eat CONT)

16. *saḥ khādati kim?* ⇔ 'is he eating?'
(He) (eats) (QP)

Thus, two different interrogative patterns in English are taken care of by different word orders. Similar situation does not arise when we examine Hindi-Sanskrit language pair and we donot find this divergence. Thus, we have:

14. *kim saḥ khādati?* ⇔ '*kyā vaha khā rahā hai?*'
(QP) (he) (eat.PR) (QP) (he) (eat) (PROG) (be.PR)

15. *saḥ kim khādati?* ⇔ *vaha kyā khā rahā hai?*
(He) (IP) (eat CONT) (He) (IP) (eat)(PROG) (be.PR)

16. *saḥ khādati kim?* ⇔ *vaha khā rahā hai kyā?*
(He) (eats) (QP) (He) (IP) (eat)(PROG) (be.PR) (IP)

In Sanskrit, word order is used to decide for definiteness for a noun. For example, consider the two sentences:

17. *bālakaḥ gṛhe asti* ⇔ 'The boy is in the house'
(boy) (house.LOC) (be.PR)

18. *gṛhe bālakaḥ asti* ⇒ 'A boy is in the house'.
(house.LOC) (boy) (be.PR)

Thus, *bālakaḥ* occurs at different positions in the sentence to show 'a definite boy' (example 17) and 'some boy' (example 18). In other words, the bare noun phrase '*bālaka*' in 17 and 18 is mapped by definite and indefinite noun phrases in English. However, the only difference between these two Sanskrit sentences is the respective positions of the subject NP and the adverbial phrase. When we look at the reverse translation of 18, we find that the nature of divergence is different. Thus, we have:

19. A boy is in the house. ⇒ *gṛhe ekaḥ bālakaḥ asti*
(house.LOC) (a) (boy) (be.PR)

On the other hand, there is no divergence between Hindi-Sanskrit language pair on this issue.

3.2 Change of Voice

In Sanskrit language, we find the use of passive voice to be very frequent, which is not so in English. We are presenting the examples below which show divergence when we go from Sanskrit to English translation. The Sanskrit sentence is in passive voice, while the corresponding sentence in English sentence is in active voice.

20a. *rāmeṇa hasitavyam* (Ram.INS)(laugh.KR)	⇒ Ram should smile. ⇒*rāmaḥ haset.* (Ram) (smile.IMPR)
21a. *kopaḥ na karaṇīyaḥ bhavatā* (anger) (not) (do.KR) (you.INS)	⇒ You should not be angry. ⇒ *tvam mā krudhya* (you) (not) (anger.IMPR)
22a. *tena khāditaḥ* (he.INS) (eat.PASS)	⇒ He ate. ⇒ *saḥ akhādat.* (he) (eat.PST)

While examining Hindi-Sanskrit language pair, we do not find similar divergence since the corresponding Hindi sentences are very close to the passive construct in Sanskrit:

20b. *rāmeṇa hasitavyam* (Ram.INS)(laugh.KR)	⇒ *rāma ko haṃsanā chāhie* (Ram.ACC) (laugh) (should)
21b. *kopaḥ na karaṇīyaḥ bhavatā* (anger) (not) (do.KR) (you.INS)	⇒ *āpako gussā nahīṃ karanā chāhie* (you.ACC) (anger) (not) (do) (should)
22b. *tena khāditaḥ* (he.INS) (eat.PASS)	⇒ *usane khāyā* (He) (eat.PST)

3.3 Gerunds and Participle Clauses

Another significant source of divergence in Sanskrit and English/Hindi can be located in the way various clauses and adjuncts are realized in different languages. First, let us consider English and Sanskrit language pair:

23. 'He is happy **to protect** the country'
⇔ *deśam* ***rakṣitvā*** *saḥ prasannaḥ bhaviṣyati.*
(country.ACC) (protect.GER) (he) (happy) (be.FU)

24. 'He came here **to protect** the country'
⇔ *deśam* ***rakṣitum*** *saḥ atra āgacchat*
(country.ACC) (protect.GER) (he) (here) (come.PST)

25a. He is not able **to walk**.
⇔ *saḥ* ***calitum*** *asamarthaḥ.*
(He) (walk.GER) (not able)

We notice that in example 23 and 24, in Sanskrit, different types of adjunct verbal clauses and complement verbal clauses are realized by different structures. In English, they are realized by an infinitive clause. The examples 24 and 25a have similar sentence construction. We now examine some sentences between Sanskrit and Hindi languages:

25b. *vaha* ***calane meṃ*** *asamartha hai* ⇔ *saḥ* ***calitum*** *asamarthaḥ.*
(He) (walk) (in) (able) (not) (be.PR) (He) (walk.GER) (not able)

26. *vaha citra* ***dekhane (ke liye)*** *āyā* ⇔ *saḥ citram* ***draṣṭum*** *āgataḥ.*
(He) (picture) (see) (for) (come.PST) He) (picture) (see.GER) (come.PST)

In the Hindi sentences in (25b-26), the adjunct verbal clauses and complement verbal clauses are realized by different structures, which in Sanskrit are mapped by a single structure. Though, for example 25b, we have a Sanskrit parallel as '*saḥ calane akuśalaḥ*' which does not present divergence.

3.4 Morphological Gaps

We take the example of causatives:

27. 'I study' ⇒ *aham paṭhāmi*
(I) (study.PR)

28. 'I make him study' ⇒ *aham tam pāṭhayāmi*
(I) (He.ACC) (teach.PR)

In the above two sentences, the form *paṭhāmi* and *pāṭhayāmi* are morphologically derived from the root *paṭh*, while the English counterpart has only one lexical verb 'study' and other is derived using the verbs such as 'get', 'make' etc, with separate argument structure. In case of Hindi-Sanskrit, no divergence is exhibited as such since in Hindi also, roots are morphologically derived:
paḍhā ⇒ *paḍhāyā*⇒*paḍhavāyā.*

3.5 Honorific

In Sanskrit, honorific features are expressed by the use of plural pronoun (as well as adjective and noun, this is crucial since the verb endings need to agree with noun) and plural verb inflections. For example, consider the sentence:

29.Respected teacher is teaching the students.
⇔ *pūjyāḥ gurucaraṇāḥ śiṣyān pāṭhayanti.*
(respected.pl) (teacher.pl) (students.ACC) (teach.PR)
⇔ *pūjya gurujī śiṣyoṃ ko paḍhāte haiṃ*
(respected) (teacher) (students) (to) (teach)

We see that in example 29, the adverb '*pūjya*', noun '*gurucaraṇa*' as well as the verb '*pāṭh*' take plural inflections in case of Sanskrit, while in hindi only the verb '*paḍhāte haiṃ*' takes the plural inflection. This divergence is caused by the socio-cultural aspect of the respective languages.

3.6 Mapping of Time

In English, the concept of a.m. vs p.m cannot be exactly mapped in Sanskrit. The example 30 shows that the time at 5 o'clock in the morning (*prātaḥkāle pañcavādane*) is denoted by a.m. in English. In example 31, the time at 11 o'clock in the morning/afternoon (*prātaḥkāle/madhyadine ekādaśavādane*) is also denoted by a.m. in English. Therefore, the term a.m. (and similarly p.m.) cannot be translated as such. One needs to examine the numbers written before and should have a built in intelligence in the translation system to handle different numbers by appropriate Sanskrit words.

When we go from Sanskrit to English translation, this divergence pattern is not exhibited since English also has more terms for periods of day than a.m. and p.m.

30. He arrived at 5 a.m. ⇒ *saḥ prātaḥkāle pañcavādane āgataḥ*
(He) (morning.LOC) (at 5 o'clock) (arrive.PST).
⇒ He came at 5 o'clock in the morning.

31. He arrived at 11 a.m. ⇔ *saḥ prātaḥkāle/madhyadine ekādaśavādane āgataḥ.*
(He) (morning/afternoon.LOC) (at 11 o'clock) (arrive.PST).

A similar situation is seen with respect to the mapping of p.m. in the examples 32-34.

32.He arrived at 3 p.m. ⇔ *saḥ aparāhne trivādane āgataḥ.*
(He) (afternoon.LOC) (at 3 o'clock) (arrive.PST).

33. He arrived at 5 p.m. ⇒ *saḥ sāyaṃkāle pañcavādane āgataḥ*
(He) (evening.LOC) (at 5 o'clock) (arrive.PST).
⇒ He came at 5 o'clock in the evening.

34. He arrived at 11 p.m. ⇒ *saḥ rātrau ekādaśavādane āgataḥ*
(He) (night.LOC) (at 11 o'clock) (arrive.PST).
⇒ He came at 11 o'clock in the night.

However, there is no divergence in case of this mapping, when we examine Hindi-Sanskrit language pair.

4 Conclusions and Discussions

Above mentioned are some of the divergence patterns that we were able to classify. We have kept in view the classification of translation divergence proposed by Dorr and some of the works on Hindi-English divergence [Sinha and Thakur, 2005] [Dave et. al., 2002]. We are in the process of identifying other such patterns. These divergence patterns will be useful in our implementation of machine translation system from English to Sanskrit language. Some of the divergence study has been useful in the current implementation of our machine translation system [Goyal and Sinha, 2008] from English to Sanskrit.

References

[Dorr, 1994] Dorr, B.: Classification of Machine Translation Divergences and a Proposed Solution. Computational Linguistics 20(4), 597–633 (1994)

[Habash and Dorr, 2002] Habash, N., Dorr, B.: Handling Translation Divergences: Combining Statistical and Symbolic Techniques in Generation-Heavy Machine Translation Technical Report, LAMP 88 (2002)

[Dave et. al., 2002] Dave, S., Parikh, J., Bhattacharya, P.: Interlingua Based English-Hindi Machine Translation and Language Divergence. Journal of Machine Translation (JMT) 17 (2002)

[Levin, 1997] Levin, B.: English Verb Classes and Alterations: A Preliminary Investigation. The MIT Press, Cambridge (1997)

[Sinha and Thakur, 2005] Sinha, R.M.K., Thakur, A.: Translation Divergence in English-Hindi MT EAMT, Budapest, Hungary (2005)

[Goyal and Sinha, 2008] Goyal, P., Sinha, R.M.K.: A Study towards English to Sanskrit Machine Translation system. SISSCL (2008)

Appendix 1

ACC: Accusative Case, **INS**: Instrumental Case, **LOC**: Locative Case,**AFF**: Affirmative Case, **CAUS**: Causative Case, **CONT**: Continuative Aspect, **CPP**: Conjunctive Participal Particle, **ET:** Determiner, **DUR**: Durative Aspect; **EW**: Echo Word, **FU**: Future Tense,**DAT**: Dative Case, **DIT**: ditransitive Case, **ERG**: Ergative Case, **GER**: Gerund, **HAB**: Habitual Aspect, **IMP**: Imperfective Aspect, **IMPR**:.Imperative Mood, **PASS**: Passive Particle, **PR**: Present Case, **INT**: Interrogative, **OPT**: Optative Mood, **QP:** Question Particle, **RP**: Relative Pronoun, **SUBJ**: Subjunctive Mood, **TRS**: Transitive, **VPRT**:Verbal Participle, **KR:** *kṛtya pratyayānta* in Sanskrit

Web Concordance of the Prakīrṇa-Prakāśa of Helārāja on the Jātisamuddeśa (3.1) of Vākyapadīya

Malhar Kulkarni and Chaitali Dangarikar

Indian Institute of Technology, Mumbai, India
malhar@hss.iitb.ac.in,
chaitali@hss.iitb.ac.in
http://www.hss.iitb.ac.in

Abstract. This article presents features of the web concordance in the form of KWIC (key word in context) index of the *Prakīrṇa-Prakāśa* of Helārāja (980 A.D.) on the *Jātisamuddesá* of *Vākyapadīya* (450 A.D.), a seminal work in Indian grammatical tradition. Apart from the original text it also takes into account variant readings in the text. This searchable concordance will be useful for the philological study of the text.

Keywords: Web concordance, KWIC Index, Sanskrit Grammar, *Prakīrṇa-Prakāśa*, *Jātisamuddesá*, *Vākyapadīya*, Bhartṛhari (450 AD), Helārāja (980 A.D.).

1 Introduction

This paper aims to present the features of the online KWIC index of the *Prakīrṇa-Prakāśa* (PP) of Helārāja[1] of the *Jāti-Samuddeśa* (JS) of *Vākyapadīya*[2] (VP) as available in Iyer's edition [5] and it's Web concordance that we have prepared during last two years. KWIC is a word index in a Key Word In Context format. The purpose of preparing the KWIC index of JS is to provide a supplementary aid to the study of VP *kāṇḍa* 3. Growing interest in the study of VP demands that the vast text of the third *kāṇḍa* should be available and accessible in the KWIC format.

Very few web concordances of Sanskrit texts are available to date. Some notable web concordances are (a) Database query to Vedic Concordance

[1] Helārāja (980 A.D.) wrote a commentary on entire VP of Bhartṛhari. The commentary on first and second *kāṇḍa*s is lost. The only work of Helārāja that is available today is his commentary on the third *kāṇḍa* of Vākyapadīya.

[2] Vākyapadīya or Trikāṇḍī (according to Aklujkar) is a wellknown treatise composed by a philosopher and grammarian called Bhartṛhari (450 AD) [2]. Rau [3] and Bhate and Kar [4] published word-index to the *kārikā* text of Vākyapadīya. There are three *kāṇḍa*s in Vākyapadīya: (1) *Brahma-kāṇḍa* or *Āgama-kāṇḍa*, (2) *Vākya-kāṇḍa*, and (3) *Pada-kāṇḍa* or *Prakīrṇa(ka)-kāṇḍa.*

A. Kulkarni and G. Huet (Eds.): Sanskrit Computational Linguistics, LNCS 5406, pp. 144–153, 2009.

[Bloomfield][3] (b) Database query to Rgvedic word concordance [Lubotsky][4] (c) Pandanus Sanskrit E-texts[5] and (d) Rāmopakhyāna index[6] is a complete concordance of a small segment of the Mahābhārata. First one is the database that represents an electronic version of M. Bloomfield's Vedic Concordance prepared by Marco Franceschini, under the supervision of Prof. Alessandro Passi, at the University of Bologna. Second one is the conversion of the first one into the STARLING-format by A. Lubotsky, October 2000, May 2005. Third one is a searchable collection of Sanskrit electronic texts is a part of the Pandanus project. At present it contains 27 *Kāvya* and *Subhāṣita* works (more than 4MB of data), all transcribed and proofread by students of the Seminar of Indian Studies (Institute of South and Central Asia, Faculty of Arts, Charles University, Prague), the work is still in progress. All these concordances are searchable databases and do not provide exact word-lists of the texts.

Plenty of e-texts on Sanskrit Grammar are available but no index is available online so far. We hope that our attempt of building the web concordance of the entire VP and PP on JS of VP (see figure 1) will be considered as a useful tool to study the terminology of Sanskrit Grammarians. The lexicon database generated by this concordance can be useful in the NLP related activities.

The reason behind choosing the text of PP on JS of VP is: Unebe prepared the index of the *Vṛtti*, a commentary on *Vākyapadīya kāṇḍa* 1 which appeared in the vol. 22 of *Saṃbhāṣā* in 2002 [6]. Unebe quoted Cardona's (1999) remarks about Professor Ashok Aklujkar's forthcoming new edition of VP which will contain index of the *Vṛtti* on the first two *kāṇḍa*s [7, p. 249-250]. We are presenting the first *Samuddeśa* of the third *kāṇḍa* of VP. Readers must note that this is an aid to the Iyer's edition of VP [5] and not an edition of VP 3.1.

The subject-matter of the third *kāṇḍa* is divided into various *Samuddeśas*. And most of the recent research works on the third *kāṇḍa* focus on a particular *Samuddeśa*, rather than studying the third *kāṇḍa* as a whole. Secondly, as Houben (1995) pointed out that the third *kāṇḍa* should be divided into two: (a) *Samuddeśas* discussing the *Jāti-pakṣa* and (b) *Samuddeśas* discussing the *Dravya-pakāṣa* [8]. According to him, in *Jāti-Samuddeśa* we find the *Jāti-padārtha-vāda*, whereas, in the *Samuddeśas* after *Dravya-Samuddeśa* emphasis is given on the *Dravya-padārtha-vāda*. The index of the individual *Samuddeśa*, will provide immense help to the researchers who study the *Samuddeśas* separately as well as for those who want to study the 3rd *kāṇḍa* in its entirety. Thinking thus we have prepared the KWIC index of the *Jāti-Samuddeśa* of the *Vākyapadīya*.

---

[3] http://www.indo-european.nl/cgi-bin/startq.cgi?flags=endnnnl&root=leiden&basename=/data/ie/bloomf

[4] http://www.indo-european.nl/cgi-bin/startq.cgi?flags=endnnnl&root=leiden&basename=/data/ie/concord

[5] http://iu.ff.cuni.cz/pandanus/electronictexts/

[6] http://sanskritlibrary.org

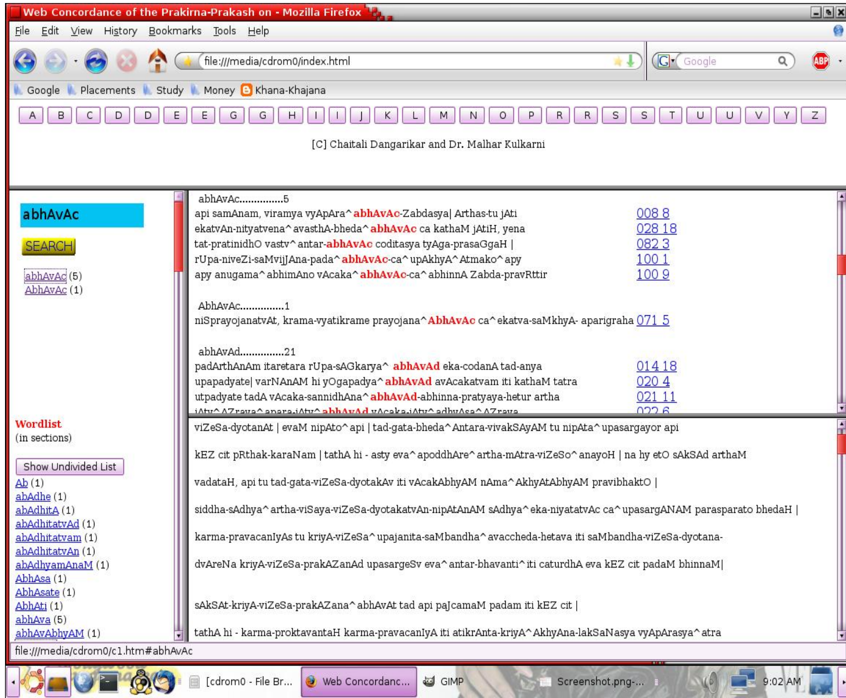


Fig. 1. Frame-based interface of Web concordance of PP on JS of VP

2 Interface of the Web Concordance

The web concordance of the *Prakīrṇa-Prakāśa* (an interface shown in figure 1) is intended for the online users. *Prakīrṇa-Prakāśa* is available as hyperlinked text. Every entry in a concordance comes with a reference which is a pointer to a particular location in the source text (text as per the Iyer's edition [5]). Researchers working on Bhartṛhari are presented with a ready-to-use concordance instead of having to construct their own, and with a browser interface which is already familiar.

Web concordance of the *Prakīrṇa-prakāśa* is prepared using the Concordance software developed by R.J.C Watt. The text is presented in the Harvard-Kyoto encoding. The only modification is "E" and "O" are used for "ai" and "au" respectively. Key Words in this concordance are sorted in the English alphabetical order.

Left panel of this web concordance contains the list of a vocabulary list and a full version of the original literary text, all hypertextually linked for ease of reference. The words are listed in the English alphabetical order. The uppermost panel contains the link to the list of keyword sorted alphabetically. In the Concordance panel, one can find a key word followed by number of occurrences. Under that the short context in which the word is used in the text can be seen.

Concordance	
Source file	G:\My Documents\KWIC-Project\JSPP-KWIC-PrintRes
Lines	4311
Words (types)	Now: 6593 When loaded: 6593
Words (tokens)	Now: 24756 When loaded: 24756
Type-token ratio	Now: 3.7549 When loaded: 3.7549
Characters	147141
Sentences	14
Words/sentence	1768.2857
Current word	AZraya
Occurrences	28
Context style	From 4 words before headword to 4 words after he

Fig. 2. Properties of the Concordance

The hyperlinked number at the right end of each occurrence is a combination of page number and line number of the hyperlinked text of the Iyer's edition [5] of *Prakīrṇa-prakāśa*. Lower panel shows the text of the *Prakīrṇa-Prakāśa* on *Jātisamuddeśa*.

2.1 Properties

Figure 2 shows information about the concordance, including the number of lines, words, characters, and sentences in the source text, the type-token ratio of the text, and the average number of words per sentence. Types are word-forms and tokens are occurrences of word-forms. The number of lines in the text of *Prakīrṇa-Prakāśa* are 4311 and there are 6593 types of words in the text. Among these 2476 are the token words. The type-token ratio is 3.6%. This ratio of types to tokens is a measure often used in quantitative stylistic analysis. The *sandhi* and *samāsa* words are split manually and hence the word-length is delimited. Following graph explains the statistical information regarding the word-lengths in the text.

The Chart's X-axis shows word length in letters. The Y-axis shows the numbers of such words. The labels at the top of each bar on the chart show the numbers of words (as on the Y-axis) but can be switched to show the percentage of the whole text which such words represent.

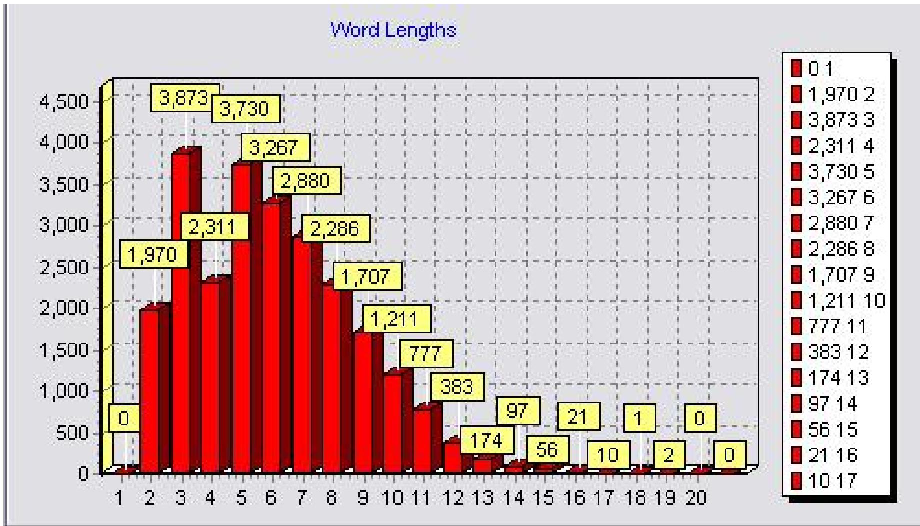


Fig. 3. Word Length Chart

3 Features of the KWIC Index

All entries in this index are arranged in alphabetical order. Each entry in this index is arranged in two columns. Each entry begins with the keywords appearing in the bold format and a number preceding to it which indicates the number of occurrences of that particular word in the text. Below this, follows a list of occurrences with the short context in two columns. First column has three sub-columns containing the number of Verse, Page no. according to Iyer's edition (1994) and line numbers at the end. In second column, occurrence of the keyword with the short context in which it appears in the JS and PP are mentioned. For example:

4 ***aṃśa***
*003 0002 0001 kriyā-sva-bhāvaḥ | //tatra ca^**aṃśa**^aṃśi-kalpanayā^apoddhāre*
*078 0002 0074 aḥ |//tatra pratipatty-artham **aṃśa**^aṃśikatayā appoddhāra-pad*
*003 0003 0001 a iti siddha-sādhya-//lakṣaṇa^**aṃśa**-dvaya-viṣayaḥ, pada^apodd*
*002 0015 0001 riyamāṇasya padasya vākyārtha^**aṃśa**-parikalpanayā^arthavata e*

Here, the number preceding the keyword ***aṃśa*** indicates the number of occurrences found in PP and JS. First entry should be read as verse 1 page 2 line 15 of Iyer's edition. // in the text indicates the beginning of the line. The compound words in the text are split into their constituent elements and these elements are shown connected by the hyphen (-). While doing so, the vowel *sandhis* are dissolved and shown as connected by ˆ . Consonant *sandhis* are shown as connected by hyphen. For example,

cāṃśāṃśikalpanayā <is shown as > *ca^**aṃśa**^aṃśi-kalpanayā*

Verbal compounds remain undissolved and mentioned as ***abhivyanakti.*** *But the **cvī** forms are dissolved.*

*087 0002 88 | na^ity āha://guṇe 'pi na^**aṅgī**-kriyate pradhāna^antara-siddhaye*
*090 0012 89 guṇe 'pi saṃkhyā//na^**aṅgī**-kriyate 'tra^iti dvi-bahuḍv*

Negative compounds like ***acoditasya*** are not dissolved as ***a-coditasya*** to avoid its confusion with the words like *a-kāra*. Similarly, compounds like *sārtham* etc. are not dissolved.

*053 0021 46 avyavacchinnāṃ śrutim āhur **akartṛkām**|//śiḍṭair nibadhyamānā*
*052 0011 46 svārtha-pratipatti-darśanād **akiñcit**-karasya^anarthakasya paratra*
*008 0007 2 rūpaṃ//padārthāv iti **sphuṭī**-kṛtam| anyathā vā^arthe*
*075 0017 71 āhuḥ: saguṇa ity etad eva **sphuṭī**-kṛtaṃ sadā^ekatvena^iti*

Readers must note that *anusvāra* is encoded following the *para-savarṇa* principle and while sorting it comes in the place of *para-savarṇa*. For example, ***aṅga*** comes after *ga* and *gh* whereas ***aṃśa*** comes in the beginning. Similarly, due to the *yaṇ-sandhis* words like ***iti*** and ***api*** should be searched twice as ***iti*** and ***ity*** (in *sandhi*) and ***api*** and ***apy.*** Same is applicable to the occurrences of the 7$^{\text{th}}$ case singular ending of the nominal words ending is *i* for example, *siddhau* and *saddhav, jātau* and *Jatāv etc.*

Additional Features of the KWIC Index

*Variant Readings (*Jāti-Samuddeśa*)* The text presented here is based on the Iyer's edition of VP 3.1 with PP on it [5]. We are thankful to Yves Ramseier for making available the digitized text of PP which saved our efforts of digitizing the full text again. After the initial proofreading we realized that the text given by Yves Ramseier needs some corrections and modifications. Apart from those modifications, we included various readings of the text of PP from the other editions, i.e., *Ambākartrī* by *Śarmā* (1991) (AK), and *Rāmakṛṣṇaśāstrī* and *Śāstrī* (1905) (A). Variant readings from the editions of *kārikā* text are also included here in the appendix. For example,

1. Verse 6 (a):- R: *svajātiḥ*; BK, AK, AL: *svā jātiḥ*. Rau mentions *sā jātiḥ* from the ms. A p. 16.
2. Verse 10 (a):- R: *yā śabdajātiśabdeṣu*; I, AK, & SK: *yā śabdajātiḥ śabdeṣu* p. 9.
3. Verse 19 (a):- AL: *pravṛttirūpāyāṃ*; I & AK: R, & SB: *pravṛttirūpā* p. 22.
4. Verse 27 (b):- BSS, I, & AK: *niṣpattau* ; Rau: *niṣpattyai* p. 36.
5. Verse 54 (a): AK and I: *karmaṇyaṅgatvam* for Rau: *karmasv aṅgatvam* p. 62.
6. Verse 57 (a): mss. *prathamaḥ.*; *prathamā* for I, AK, R: *prathamaṃ*
7. Verse 60 (c):- AK & I: *kṣāyāṃ* for Rau: *kṣāyā* p. 67.
8. Verse 62 (d): AK: *evam* for Rau & I: *eva* p. 68.

9. Verse 63 (a):- I: *ekena cet pra* for Rau & AK: *ekena ca pra* p. 69.
10. Verse 68 (b):- Rau mentions eight different readings from various mss. as *pāradharmohataste , pāradharmmotvaste, pāradharmoyataste , pāravataste, pāraevāste , pāradharmmataste , pāradharmmotaste , and pāravātaste.* Rau considers *saṃkhyāvyāpāradharmo* as one compound word whereas Iyer and AK splits it as *saṃkhyā vyāpāradharmo.* p. 72.
11. Verse 79 (a):- Rau mentions 5 different readings: *vyaktiśaktes samāpannā, vyaktiśaktes samāsanno, vyaktiśakis samāsanno, dravyaśakter yathāsannā and dravyśakter yathāsattā.* p. 81.

In the context of the Index Raus' readings [9] are indicated by using an * followed by the abbreviation R in <. . . >. For example,

*103 0003 *108 ‖105<R 109>‖//ghaṭa-jñānam **iti** jñānaṃ ghaṭa^ādy^ākāraṃ j*

Here, 105<*R 109> indicates that verse 105 of Iyer's edition [5] is recorded as verse 109 in Rau's edition [9] edition of Vākyapadīya. It means that the verse no.109 in Rau's edition is recorded as verse no.105 in Iyer's edition.

*Variant Readings (*Prakīrṇa-Prakāśa*):* KWIC index also lists the variant readings found in the text of *Prakīrṇa-Prakāśa* rendered by K. A. Subramania Iyer and Raghunath Sharma.

1. AK: *ruciraḥ* for I: *ruciraṃ* p.1:3.
2. AK: *nivartana* for I: *nirvartana* p.2:14.
3. mss. M *°upakalpita°* for I and AK: *°utkalita°* p.3:1.
4. AK: *daṇḍaka* after *iti* in 24b p.15:1.
5. AK: *na śrautārthatyāgaḥ pratinidhau* | for I: *na śrautārthatyāgaḥ* | [25b] *pratinidhau naitaṇnyāyyaṃ*| p.15:10.
6. AK: *śabdo'vastucintāṃanusarati* for I: *śabdo vastucintāṃanusarati* p.15:11.
7. AK: *pratipādyate* for I: *pratipadyate* p. 17:11.
8. AK: | p. 18:1.
9. AK: *°kṣaṇaḥ* for I: *°lakṣaṇaḥ* p. 18:3.
10. AK: *°abhivyañjane iti* for I: *°abhivyañjana iti* p. 18:4.
11. AK: *°prasaṅga ityāśaṅkyāha* for I: *°prasaṅgaityāśaṅkyāha* p. 22:13.
12. AK: in the main text *°avabhāitvābhāt* but *°avabhāitvābhāvāt* in the *Ambākartrī* for I: *°avabhāitvābhāvāt.* p. 24:3.
13. AK: *jāti-vyatiriktayā* , I: *jāti vyatiriktayā*; and BBS: *jātir vyatiriktayā* p. 25:11.
14. AK: *vastu san-nābhidhīyate*; I: *vastu-san-nābhidhīyate* p. 25:14.
15. AK & I: *yathā*; Ramseier: *tathā* p. 26:12.
16. AK & I: *sattva-guṇāḥ* | *tathā*; Ramseier: *sattvaguṇās tathā* p. 27:3.
17. AK: *gotvākāraḥ prasūyate*; I: *gotvākāraḥ pratyayaḥ prasūyate* p. 27:16.

The text of Iyer's edition [5] of PP on 106 verses of JS has 105 pages, 4596 lines and 7175 words. Among these 24409 words, only 7175 words are unique words.

We hope and believe that this work will prove useful for Indologists all over the world working in the area of Language studies. We hereby attach a few sample pages of the said KWIC index (after converting HK encoding into the roman transliteration). The web concordance of the PP on JS of VP will be made available on the IIT website at appropriate time.

References

1. Potter, K.H.: Helārāja. In: Potter, K. (ed.) The Philosophy of Greammarians. Encyclopedia of Indian Philosophy, vol. V, pp. 193–197. Motilal Banarasidass, Delhi (1990)
2. Aklujkar, A.: Two textual studies of Bhartṛhari. In: Bhate, S., Bronkhorst, J. (eds.) Bhartṛhari: Philosopher and Grammarian, Proceedings of the First International Conference, Bhartṛhari, Delhi, First Indian edn., Motilal Banarsidass Pub. (1994) (Origenally published, 1993)
3. Rau, W.: Bhartṛharis Vākyapadīya Vollständiger Wortindex zu den *mūlakārikās*. Franz Steiner Verlag, Stuttgart (1988) (Akademie der Wissenschaften und der Literatur, Abhandlungen der Geistes- und Sozialwissenschaftlichen Klasse, Jahrgang 1988, n°11) (in German)
4. Bhate, S., Kar, Y.: Word Index to the Vākyapadīya of Bhartṛhari. Eastern Book Linkers, Delhi (1992) (Together with the complete text of the Vākyapadīya. Follows roughly Rau's 1977 edition, but in Devanāgarī)
5. Iyer, K.A.S.: Vākyapadīya of Bhartṛhari with the Prakīrṇa-Prakāśa of Helārāja, Kāṇḍa III Part i. Deccan Collage, Pune (1994)
6. Unebe, T.: KWIC index to the Vākyapadīya, kāṇḍa 1. Nagoya Studies in Indian Culture and Buddhism. Saṃbhāṣā 22, 1–239 (2002)
7. Cardona, G.: Recent Research in Pāṇinian Studies. Matilal Banarasidass, Delhi (1999)
8. Houben, J.E.M.: The Saṃbandha-samuddeśa (Chapter on Relation) and Bhartṛhari's Philosophy of Language. Egbert Forsten (1995)
9. Rau, W.: Bhartṛhari's Vākyapadīya: Die Mūlakārikās nach den Handschriften heraugegeben und mit einem Pāda-Index versehen von Wilhelm Rau. Reprint edn. Franz Steiner, Wiesbaden (2002)

Abbreviations Used

A *Rāmakṛṣṇaśāstrī* and *Śāstrī* (1905)
AK *Ambākartrī* by *Śarm*ā (1991)
I Iyer's edition of *Vākyapadīya of Bhathari*
JS *Jāti-Samuddeśa*
PP *Prakīrṇa-Prakāśa*
R Rau's edition of *Vākyapadīya of Bhatṛhari*
VP *Vākyapadīya*

About Authors

Dr. Malhar Kulkarni, Associate Professor, Department of Humanities and Social Sciences, Indian Institute of Technology, Powai, Mumbai, 400076.

Chaitali Dangarikar, Research Scholar, Department of Humanities and Social Sciences, Indian Institute of Technology, Powai, Mumbai, 400076.

A Sample of the KWIC Index

4 ***aṃśa***

*003 0002 0001 kriyā-sva-bhāvaḥ | //tatra ca^* ***aṃśa*** *^aṃśi-kalpanayā^apoddhāre*

078 0002 0074 aḥ |//tatra pratipatty-artham ***aṃśa*** *^aṃśikatayā appoddhāra-pad*

*003 0003 0001 a iti siddha-sādhya-//lakṣaṇaˆ**aṃśa**-dvaya-viṣayaḥ, padaˆapodd*

*002 0015 0001 riyamāṇasya padasya vākyārthaˆ**aṃśa**-parikalpanayāˆarthavata e*

*1 **aṃśaḥ***

*003 0004 0001 hedāt kārakaˆātmā siddha-rūpoˆ**aṃśaḥ** | yady api ca nāma-padā*

*1 **aṃśasya***

*068 0021 0062 jātiḥ padārtho bādhyeta, jātyˆ**aṃśasya**ˆajātitvād//iti prakṛty*

*1 **aṃśā***

*047 0005 0040 ātrāḥ kalāḥ parikalpitā bhāgā **aṃśā** gotvaˆādi-sāmānya-viśeṣāḥ*

*1 **aṃśi***

*003 0002 0001 -sva-bhāvaḥ | //tatra caˆaṃśaˆ**aṃśi**-kalpanayāˆapoddhāre kārak*

*1 **aṃśikatayā***

*078 0002 0074 /tatra pratipatty-artham aṃśaˆ**aṃśikatayā** appoddhāra-padārtha*

*1 **aṃśo***

*003 0002 0001 āre kārakaˆātmā kriyāˆātmā caˆ**aṃśo** vibhāgaˆarha iti siddha-*

*1 **akartṛ***

*090 0006 0089 yasyaˆasti kartṛ-saṃjñakasyaˆ**akartṛ**-saṃjñakasya ca tasyaˆub*

*1 **akartṛkām***

*053 0021 0046 im-avyavacchinnāṃ śrutim āhur **akartṛkām**|//śiṣṭair nibadhyamān*

*1 **akarmakebhyaḥ***

*090 0008 0089 vaṃ //laḥ karmaṇi ca bhāve caˆ**akarmakebhyaḥ** (P. 3.4.69)//it*

*1 **akāra***

*073 0004 0068 maḥ, tena liṅgena gamyata ity **akāra**-praśleṣaṃ ke cid vyācakṣ*

*1 **akāraṇāni***

*038 0001 0028 virodhaḥ| tathā caˆāhuḥ:///naˆ**akāraṇāni** vidhiṃ bādhante prat*

*1 **akāri***

*033 0009 0019 atirikta-sāmānya-kalpanāˆevam **akāri**ˆity atra tātparyaˆarthaḥ*

*2 **akiñcit***

*052 0011 0046 ṃ svārtha-pratipatti-darśanād **akiñcit**-karasyaˆanarthakasya p*

*088 0023 0089 antryaṃ sarveṣāṃ eva vidyate, **akiñcit**-karasya kriyāˆaṅga-bhā*

*1 **akramatā***

*020 0019 07-8 stavaḥ kramaḥ | pratyāyane tv **akramatā**ˆeva| śabdaˆācchuritat*

*1 **akṣareṇa***

*018 0012 0006 avat sphoṭa-tattvam| prathamaˆ**akṣareṇa** hi jāter ābhāsa-mātra*

*4 **akhaṇḍa***

*002 0003 0001 ada-vyutpattir-vākya-vādinām, **akhaṇḍa**-pada-//vyutpattāv iva*

*079 0002 0075 katayā//pratipāditaḥ | vākyād **akhaṇḍa**-pratibhāˆutpattāv apod*

*078 0015 0074 stutaḥ | tathā ca śruty-ādīnyˆ**akhaṇḍa**-vākya-pakṣe naˆupapady*

*002 0003 0001 dhyā//pṛthak padaṃ niṣkṛṣya | **akhaṇḍa**-vākya-vyutpattāv upāya*

*1 **akhaṇḍe***

*015 0002 0005 ˆadhyavāpaḥ/śabda-vyāpāraḥ |//**akhaṇḍe**ˆapi hi vākyārtha-nayeˆ*

*1 **agata***

*074 0021 0070 vādāt saṃmārge pradhāna-velāvˆ**agata**-saṃkhyākānām evaˆavadhār*

*1 **agatyā***

*097 0007 0096 ṃ sva-saṃvijñāna-padaˆabhāvād **agatyā** bhedaˆabheda-śabdābhyāṃ*

*1 **agṛhyamāṇa***

*039 0018 30-31 śrotra^indriya-//kāryam| evam **agṛhyamāṇa**-sva-bhāvā vyaktayaḥ*

*2 **agṛhyamāṇasya***

*040 0001 30-31 yaṃ///śabda^artha ity arthaḥ| **agṛhyamāṇasya**^api tadānīṃ vyak*

*040 0002 30-31 ya^avasāya^avirodhāt saṃpraty^**agṛhyamāṇasya**^api dṛṣṭaṃ pratī*

*1 **agni***

*064 0009 0056 atāt| tathā hi "āgneyam ajam-**agni**-ṣṭoma ālabheta" ity uktvā*

*2 **agnau***

*054 0013 0046 athā^āditye tejaḥ, dāhakatvam **agnau**, śaityam apsu| tathā jñāna*

*088 0012 0089 tarhi yajeḥ prayogo dṛśyate, "**agnau** suṣṭhu yajata" iti| atra*

Author Index